322-CD-002-002

EOSDIS Core System Project

Release A
Integration & Test Procedures for the ECS Project
Volume 1: CSMS

Preliminary

August 1996

Hughes Information Technology Systems
Upper Marlboro, Maryland

Release A Integration & Test Procedures for the ECS Project Volume 1: CSMS

Preliminary

August 1996

Prepared Under Contract NAS5-60000
CDRL Item 055

SUBMITTED BY

Parag N. Ambardekar	7/30/96
Parag Ambardekar, Release A CCB Chairman EOSDIS Core System Project	Date

Hughes Information Technology Systems
Upper Marlboro, Maryland

322-CD-002-002

This page intentionally left blank.

Preface

This document is a contract deliverable with an approval code 2. As such, it does not require formal Government approval, however, the Government reserves the right to request changes within 45 days of the initial submittal. Once approved, contractor changes to this document are handled in accordance with Class I and Class II change control requirements described in the EOS Configuration Management Plan, and changes to this document shall be made by document change notice (DCN) or by complete revision.

Any questions should be addressed to:

Data Management Office
The ECS Project Office
Hughes Information Technology Systems
1616 McCormick Drive
Upper Marlboro, Maryland 20774-5372

This page intentionally left blank.

Abstract

The Release A Integration and Test Procedures satisfies the requirements for CDRL items 055, DID 322/DV2 (Release Integration & Test Procedures), as specified in the Statement of Work, as a deliverable under the Earth Observing System Data and Information System (EOSDIS) Core system (ECS) contract number NAS5-60000.

The Release A Integration and Test Procedures contains two subdocuments. Volume 1: CSMS Procedures, contains the test cases and procedures primarily to verify the CSMS related Level 3 and Level 4 requirements. Volume 2: SDPS Procedures, contains the test cases and procedures primarily to verify the SDPS related Level 3 and Level 4 requirements. In some cases CSMS or SDPS requirements have been mapped to test cases in Volumes 2 or 1, respectively, to support a more efficient verification approach. This document should be considered as a whole, the CSMS and SDPS volumes serve only to partition the document into manageable sized subdocuments for publication.

This is the Preliminary Version of the Release A Integration and Test Procedures, Volume 1: CSMS Procedures. This subdocument contains the test cases and selected test procedures to demonstrate CSMS functionality and performance as specified in the Level 3 and 4 requirements. This document is intended to provide insight into the consolidated and refined test cases and a sample of the test procedures. This document will be incrementally updated as the remaining procedures are added, refined and delivered according to the schedule in Section 1.4

Keywords: integration, test, I&T, build, thread, Release-A, CSMS, SDPS, ECS, CSS, MSS

This page intentionally left blank.

Change Information Page

List of Effective Pages	
Page Number	**Issue**
Title	Preliminary for TRR
iii and xii	Preliminary for TRR
1-1 and 1-2	Preliminary for TRR
2-1 through 2-4	Preliminary for TRR
3-1 through 3-8	Preliminary for TRR
4-1 through 4-540	Preliminary for TRR
A-1 through A-134	Preliminary for TRR
B-1 through B-28	Preliminary for TRR
AB-1 through AB-6	Preliminary for TRR

Document History			
Document Number	**Status/Issue**	**Publication Date**	**CCR Number**
322-CD-002-001 414-CD-002-001	Preliminary	November 1995	95-0851
322-CD-002-002	Preliminary for TRR	August 1996	96-0850

This page intentionally left blank.

Contents

Preface

Abstract

1. Introduction

1.1 Identification 1-1
1.2 Scope 1-1
1.3 Purpose 1-1
1.4 Status and Schedule 1-1
1.5 Organization 1-2

2. Related Documents

2.1 Parent Documents 2-1
2.2 Applicable Documents 2-1
2.3 Information Documents 2-2
2.3.1 Information Documents Referenced 2-2
2.3.2 Information Documents Not Referenced 2-3

3. Release A Integration and Test Overview

3.1 Release A I&T Overview 3-1
3.1.1 Functional Overview 3-1
3.2 Release I&T Organization Testing Approach 3-2
3.2.1 Release A I&T Testing 3-2
3.2.2 Build/Thread Methodology 3-3
3.3 I&T Test Verification 3-4

3.3.1 Verification Methods 3-4
3.3.2 Post Test Analysis 3-4
3.3.3 Regression Testing 3-5

4. CSMS Release A Test Procedures

4.1 Basic Communications Environment Tests 4-1
4.1.1 Internetworking Thread (TC001) 4-1
4.1.2 DCE Security Authentication Thread Test (TC002) 4-28
4.2 Basic Host/Application Management Tests 4-61
4.2.1 MIB/Extensible Host Thread (TC003) 4-62
4.2.2 MIB/Extensible Agent - Application 1 Thread (TC004) 4-64
4.2.3 Log File Management Thread (TC005) 4-79
4.2.4 Basic Host/Application Management Build (BC002) 4-81
4.3 Configuration Management Tests 4-87
4.3.1 Trouble Ticketing Thread (TC007) 4-87
4.3.2 Software CM Thread (TC008) 4-97
4.3.3 Physical Configuration Management Thread (TC009) 4-112
4.4 Communications Infrastructure 1 Tests 4-122
4.4.1 Message Passing One Thread (TC010) 4-122
4.4.2 Directory Naming Service Thread (TC011) 4-141
4.4.3 Communication Infrastructure 1 Build (BC003) 4-162
4.5 External Interface Communications Tests 4-169
4.5.1 User Account Management Thread (TC012) 4-169
4.5.2 FTP/Kerberos FTP Thread (TC013) 4-172
4.5.3 Client/Server Gateways Thread (TC014) 4-180
4.5.4 Security Authorization Thread (TC015) 4-186
4.5.5 External Interface Communications Build (BC004) 4-208
4.6 Application Support Services Tests 4-218
4.6.1 Time Service Thread (TC016) 4-218
4.7 Application Management Service Tests 4-232
4.7.1 MSS Lifecycle Services Thread (TC017) 4-232

4.7.2 Discover/Monitor Thread (TC018) 4-263
4.7.3 MIB/Extensible Agent - Application 2 Thread (TC019) 4-271
4.7.4 Process Framework Thread (TC040) 4-280
4.7.5 Application Management Services Build (BC006) 4-290
4.8 Management Services 1 Tests 4-302
4.8.1 Performance Management Thread (TC020) 4-302
4.8.2 Hardware Fault Management Thread (TC021) 4-313
4.8.3 Management Data Access I Thread (TC022) 4-326
4.8.4 Accountability Thread (TC023) 4-332
4.8.5 Change Request Manager Thread (TC024) 4-339
4.8.6 Management Services 1 Build (BC007) 4-367
4.9 Communications Infrastructure 2 Tests 4-386
4.9.1 Message Passing Two Thread (TC025) 4-386
4.9.2 Communications Infrastructure 2 Build (BC008) 4-401
4.10 Management Services 2 Tests 4-405
4.10.1 Software Fault Management Thread (TC027) 4-405
4.10.2 Security Management Thread (TC028) 4-409
4.10.3 Operator Role Management Thread (TC029) 4-416
4.10.4 Management Data Access Data Extraction Thread (TC030) 4-420
4.10.5 Baseline Manager Thread (TC031) 4-435
4.10.6 Management Services 2 Build (BC009) 4-447
4.11 Communications Infrastructure Three Tests 4-477
4.11.1 Electronic Mail (E-Mail) /BBS Thread (TC032) 4-477
4.11.2 Virtual and X-Terminal Support Thread (TC033) 4-489
4.11.3 Managed ftp Thread (TC034) 4-492
4.11.4 Communications Infrastructure 3 Build (BC010) 4-498
4.12 SMC Tests 4-519
4.12.1 SMC Ground Events Scheduling Thread (TC036) 4-519
4.12.2 SMC Security Management Thread (TC037) 4-520
4.12.3 SMC Trouble Ticketing Thread (TC038) 4-523
4.12.4 SMC Configuration Management Thread (TC039) 4-523
4.12.5 SMC Build (BC011) 4-528

Figures

3.2-1 Release A Build/Thread Diagram 3-7

Tables

3.1-1 CSMS Release A CIs 3-1

Appendix A. Requirements Verification Matrix

Appendix B. Test Plan to Test Procedure Matrix

Abbreviation and Acronyms

1. Introduction

1.1 Identification

This document is submitted as required by CDRL items 055, DID 322/DV2 whose requirements are specified as a required deliverable under the Earth Observing System (EOS) Data and Information System (EOSDIS) Core System (ECS), Contract (NAS5-60000).

1.2 Scope

The Release A Integration and Test Procedures satisfies the requirements for CDRL items 055, DID 322/DV2 (Release Integration & Test Procedures), as specified in the Statement of Work, as a deliverable under the Earth Observing System Data and Information System (EOSDIS) Core system (ECS) contract number NAS5-60000.

The Release A Integration and Test Procedures contains two subdocuments. This subdocument, Volume 1: CSMS Procedures, contains the test cases and procedures primarily to verify the Communications and Systems Management Segment (CSMS) related Level 3 and Level 4 requirements. Volume 2: SDPS Procedures, contains the test cases and procedures primarily to verify the Science Data Processing Segment (SDPS) related Level 3 and Level 4 requirements. In some cases CSMS or SDPS requirements have been mapped to test cases in Volumes 2 or 1, respectively, to support a more efficient verification approach. This document should be considered as a whole, the CSMS and SDPS volumes serves only to partition the document into manageable sized subdocuments for publication.

This document reflects the Technical Baseline submitted via contract correspondence no. ECS 194-00343.

1.3 Purpose

This is the Preliminary Version of the Release A Integration and Test Procedures, Volume 1: CSMS Procedures. This subdocument contains the test cases and selected test procedures to demonstrate CSMS functionality and performance as specified in the Level 3 and 4 requirements. This document is intended to provide insight into the consolidated and refined test cases due to ECS organizational and implementation plan changes and a sample of the test procedures. This document will be incrementally updated as the remaining procedures are added, refined and delivered according to the schedule in Section 1.4.

1.4 Status and Schedule

This is the Preliminary Version of the Release A Integration and Test Procedures. This document contains the Segment and System test cases revised due to ECS organizational and implementation plan changes. A mapping of the test cases from the approved Release A SDPS, CSMS, System Integration & Test Plans (319-CD-004-003, 319-CD-002-002, and 402-CD-002-

002) to the test cases in this document are provided in Appendix B. Once approved this document supersedes the respective Segment and System Integration & Test Plans. Future changes to the Release A test cases will be reflected in this document.

Two more formal releases are scheduled for this document. At the Test Readiness Review (TRR) the document is released containing test procedures for all segment level test cases. Segment level test cases are defined as test that primarily verify Level 4 requirements. Three months prior to the Release Readiness Review (RRR) the document is released containing test procedures for all system level test cases. System level test cases are defined as test cases that primarily verify Level 3 requirements.

Staged updates to test cases and procedures are delivered at the start and during the test rehearsals, as needed, for each set of tests described in this document. These updates are considered working versions of this document and are released informally on an "as required" basis.

1.5 Organization

This document, which is based on Release A requirements, is organized into four chapters:

Section 1	Introduction, contains the identification, scope, purpose and objectives, status and schedule, and document organization.
Section 2	Related Documents, provides a bibliography of parent, applicable and reference documents for the Release A Segment and System Integration and Test Procedures Document.
Section 3	Release Integration and Test Overview, describes the process used to integrate and test the SDPS and CSMS Segments.
Section 4	Test Procedures, describes the specific segment and system level thread and build test cases and the step by step test procedures, which are used to verify the SDPS and CSMS functionality.
Appendix A	Requirements Verification Matrices. Contains the requirements traceability matrices, mapping test cases to Release A Level 3 and 4 requirements.
Appendix B	Test Plan to Test Procedure Matrices, Contains the mapping of the test cases from the approved Segment and System Test Plans to the test cases in this document.
Acronyms List	Contains a listing of abbreviations and acronyms used in this Document.

2. Related Documents

2.1 Parent Documents

The parent documents are the documents from which volume's scope and content are derived.

101-CD-001-004	Project Management Plan for the EOSDIS Core System
194-107-MG1-XXX	Level 1 Master Schedule for the ECS Project
194-201-SE1-001	Systems Engineering Plan for the ECS Project
301-CD-002-003	System Implementation Plan for the ECS Project
319-CD-004-003	CSMS Integration and Test Plan for the ECS Project, Volume 2: Release A
319-CD-005-002	SDPS Integration and Test Plan for the ECS Project, Volume 2: Release A
402-CD-002-002	System Integration and Test Plan for the ECS Project, Volume 2: Release A, Final
501-CD-001-004	Performance Assurance Implementation Plan for the ECS Project
423-41-01	Goddard Space Flight Center, EOSDIS Core System (ECS) Statement of Work
423-41-02	Goddard Space Flight Center, Functional and Performance Requirements Specification (F&PRS) for the Earth Observing System Data and Information System (EOSDIS) Core System (ECS)
423-41-03	Goddard Space Flight Center, EOSDIS Core System (ECS) Contract Data Requirements Document

2.2 Applicable Documents

The following documents are referenced within this volume, or are directly applicable, or contain policies or other directive matters that are binding upon the content of this volume.

194-207-SE1-001	System Design Specification for the ECS Project
194-219-SE1-018	Interface Requirements Document Between EOSDIS Core System (ECS) and Tropical Rainfall Measuring Mission (TRMM) Ground System

194-401-VE1-002	Verification Plan for the ECS Project, Final
404-CD-001-001	Procedure for Control of Unscheduled Activities During Verification
409-CD-001-004	ECS Overall System Acceptance Test Plan for Release A, Final
194-415-VE1-002	Acceptance Testing Management Plan for the ECS Project, Final

2.3 Information Documents

2.3.1 Information Documents Referenced

The following documents are referenced herein and, amplify or clarify the information presented in this document. These documents are not binding on the content of this volume.

102-CD-001-004	Development Configuration Management Plan for the ECS Project
193-103-MG3-001	Configuration Management Procedures for the ECS Project
305-CD-004-001	Overview of Release A SDPS and CSMS System Design Specification for the ECS Project
305-CD-005-001	Release A SDPS Client Subsystem Design Specification for the ECS Project
305-CD-006-001	Release A SDPS Interoperability Subsystem Design Specification for the ECS Project
305-CD-007-001	Release A SDPS Data Management Subsystem Design Specification for the ECS Project
305-CD-008-001	Release A SDPS Data Server Subsystem Design Specification for the ECS Project
305-CD-009-001	Release A SDPS Ingest Subsystem Deign Specification [for the ECS Project]
305-CD-010-001	Release A SDPS Planning Subsystem Design Specification for the ECS Project
305-CD-011-001	Release A SDPS Data Processing Subsystem Design Specification for the ECS Project
305-CD-012-001	Release A CSMS Communications Subsystem Design Specification for the ECS Project
305-CD-013-001	Release A CSMS Systems Management Subsystem Design Specification for the ECS Project, Final

305-CD-014-001	Release A GSFC DAAC Design Specification for the ECS Project
305-CD-015-001	Release A LaRC DAAC Design Specification for the ECS Project
305-CD-016-001	Release A MSFC DAAC Design Specification for the ECS Project
305-CD-017-001	Release A EDC DAAC Design Specification for the ECS Project
305-CD-018-001	Release A Data Dictionary for Subsystem Design Specification for the ECS Project
305-CD-019-001	Release A System Monitoring and Coordination Center Design Specification for the ECS Project

2.3.2 Information Documents Not Referenced

The following documents, although not referenced herein and/or not directly applicable, do amplify or clarify the information presented in this document. These documents are not binding on the content of the Release A Integration and Test Plan and Procedures.

104-CD-001-004	Data Management Plan for the ECS Project
193-105-MG3-001	Data Management Procedures for the ECS Project

This page intentionally left blank.

3. Release A Integration and Test Overview

This section contains an overview of the approach taken by the Release A Integration and Test organization to ensure complete and thorough testing at the segment and system levels. Included is information concerning the I&T approach, verification activities, and responsibilities.

3.1 Release A I&T Overview

3.1.1 Functional Overview

ECS is comprised of three segments, each comprised of various subsystems. The three segments are the Flight Operations Segment (FOS), Science Data Processing Segment (SDPS) and Communications and Systems Management Segment (CSMS). Each of these segments are decomposed into subsystems and the subsystems are composed of CIs. This document provides the procedures for testing the design and implementation of the CSMS and SDPS CIs and their integration into ECS subsystems for Release A.

This subdocument, Volume 1: CSMS Procedures, focuses on the integration of CSMS subsystems. CSMS is responsible for the interconnection of users and service providers, the transfer of information between ECS components and systems management. CSMS is also responsible for supporting and providing interoperability for the Science Data Processing Segment (SDPS) and the Flight Operations Segment (FOS). CSMS contains three internal subsystems: Communications Subsystem (CSS), Inter networking Subsystem (ISS) and the Systems Management Subsystem (MSS). CSS is a collection of services that are responsible for providing flexible interoperability and information transfer between clients and servers. ISS is a layered stack of communications services corresponding to layers 1-4 of the OSI-RM. MSS is made up of a collection of applications that are responsible for the management of all ECS resources, including all SDPS, FOS, ISS and CSS components.

The CIs for CSMS at Release A are listed in Table 3.1-1. Included are CI names and CSMS subsystems. A short description of each CI is given following Table 3.1-1.

Table 3.1-1. CSMS Release A CIs

CI	Subsystem Superclass
Management Software CI (MCI)	MSS
Management Logistics CI (MLCI)	MSS
Management Agent CI (MACI)	MSS
Management Hardware CI (MHCI)	MSS
Distributed Computing Software CI (DCCI)	CSS
Distributed Communications Hardware CI (DCHCI)	CSS
Internetworking CI (INCI)	ISS
Internetworking Hardware CI (INHCI)	ISS

Management Software CI (MCI). The Management Software CI includes both Enterprise System Monitor and Local System Manager Configurations.

Management Logistics CI (MLCI). The Management Logistics CI includes both the managing of the Enterprise Logistics Manager and Domain Logistics Manager Configurations.

Management Agent CI (MACI). The Management Agent CI includes the agent specializations. An agent is the interface to a managed object. An agent system is a device, which has the responsibility of performing network management operations requested by the manager.

Management Hardware CI (MHCI). The Management Hardware CI includes: local management of ECS sites enterprise-wide (ECS-wide) monitoring and coordination, shared workstation and server pool for enterprise and domain managers, as well as enterprise-wide (ECS-wide) monitoring and coordination.

Distributed Computing Software CI (DCCI). The Distributed Computing Software CI includes the client and server configurations. This CI includes the DCE COTS package, which is the baseline distributed computing technology chosen through Release B.

Distributed Communications Hardware CI (DCHCI). The Distributed Communications Hardware CI includes the communications servers, such as; E-mail server, directory server, security server, time server, trader server and bulletin board (user registration/toolkit distribution) server.

Internetworking CI (INCI). The Internetworking CI includes the COTS implementations of the communication protocols reserved by network nodes. This includes protocols and standards for layer four and below of the OSI/ISO reference model (e.g., TCP, IP, OSPF, RIP).

Internetworking Hardware CI (INHCI). The Internetworking Hardware CI includes all COTS network devices and cabling: routers, hubs, switches, and test equipment. It does not include host interface cards.

3.2 Release I&T Organization Testing Approach

3.2.1 Release A I&T Testing

The Release A I&T organization integrates and verifies SDPS and CSMS CI functionality on an incremental basis. As incremental integration and testing proceeds, larger portions of the segments are assembled. The integration focuses on integrating functionally related components rather than on the structural decomposition achieved through the design process. As such, components from multiple CIs or segments may be integrated and tested early in the integration process. Therefore there is no strict delineation between CI, Segment, or System Integration and Test. An attempt has been made to delineate SDPS from CSMS and Segment from System Testing based on the emphasis of the requirement verification activities.

As unit testing on software and hardware items is completed, the I&T organization incrementally assembles lower-level functionality into progressively higher levels until a Release

is completely integrated and tested. Functional components that are integrated are threads, and the result of combining threads is a build. Functional testing verifies Level 3 and 4 functional requirements.

3.2.2 Build/Thread Methodology

The build/thread concept, which is based on the incremental aggregation of functions, is used to plan I&T activities. A thread is the set of components (software CIs, hardware and data) and operational procedures that implement a function or set of related functions. Threads are tested individually to facilitate Level 4 requirements verification and to isolate software problems. A build is an assemblage of threads to produce a gradual buildup of segment capabilities. This orderly progression of combining lower level software and/or hardware items to form higher level items with broader capability is the basis of CSMS and SDPS Release A integration.

Earlier Threads and Builds focus on demonstration of selected segment functionality and verification of primarily Level 4 requirements and are thus considered segment threads and builds. Later integration activities, primarily builds, demonstrate end-to-end system level functionality and verify primarily Level 3 requirements, thus they are considered system builds.

Build/thread diagrams are developed for each release. The build/thread diagram for Release A is presented in Figure 3.2-1. Threads and builds are defined by examining CIs, Level 4 requirements, the release implementation plan and segment/element design specifications. The Release I&T organization, with support from the development community, logically groups the release into functional categories divided along noticeable boundaries. These categories are the basis for threads. Threads are combined to define builds. Builds include several integrated threads and/or other builds functions. The build/thread diagram for each release acts as a framework for test case definition. From each build and thread on the diagram, test cases are developed. These test cases provide the basis for development of step-by-step test instruction to be documented as test procedures.

Volume 1 of this document contains the test cases for the threads and builds on Figure 3.2-1 that focus on CSMS functionality. These threads and build explicitly integrate and test the functionality specified by most of the CSMS Level 3 and 4 requirements. Some CSMS requirements, such as external interfaces, are mapped to test threads or builds in Volume 2 to improve the efficiency if verification approach.

Volume 2 of this document contains the test cases for the threads and builds on Figure 3.2-1 that focus on SDPS functionality. In addition to the explicit SDPS functionality verified in these threads and builds, they also integrate and test many CSMS components. Explicit integration activities between SDPS and CSMS functional components are indicated by dashed lines and portions of the SMC and Release A DAAC Builds will be designed to run concurrently. In addition all SDPS threads and builds have implicit integrations with CSMS components, such as CSS communication services, that are not shown on the figure.

Concurrent integration and verification of CSMS components in both the CSMS and SDPS threads and builds ensures these underlying critical components are thoroughly tested and improves system reliability. Isolated testing of CSMS functionality described in Volume 1 provides intensive testing and requirements verification of these service oriented components. Early integration of CSMS components with the SDPS applications that rely on their services reduces integration risk and driver development costs.

3.3 I&T Test Verification

The following sections define responsibilities and activities of the I&T organization. I&T verification includes definition of verification methods, post test analysis, regression testing, and verification resources.

3.3.1 Verification Methods

The four verification methods used for I&T include: inspection, analysis, demonstration, and test. As defined in the ECS Verification Plan (ECS document number 194-401-VE1-002).

a. Inspection. The visual, manual examination of the verification item and comparison to the applicable requirement or other compliance documentation, such as engineering drawings.

b. Analysis. Technical or mathematical evaluation based on calculation, interpolation, or other analytical methods.

c. Demonstration. Observation of the functional operation of the verification item in a controlled environment to yield qualitative results without the use of elaborate instrumentation or special test equipment.

d. Test. A procedure or action taken to determine under real or simulated conditions the capabilities, limitations, characteristics, effectiveness, reliability, or suitability of a material, device, system, or method.

Each requirement is verified by one or more of these methods. A requirements matrix mapping Release A segment and system level requirements to Release A test cases, is provided in Appendix A of this document.

3.3.2 Post Test Analysis

Post-test analysis includes data reduction and comparison of actual results against expected results. Post test analysis required for I&T is performed by the I&T organization with support from system engineering or development organizations and the user communities when appropriate. Methods for performing post-test analysis are documented in the Segment/Element Integration and Test Procedures on a test by test basis. Results of post-test analysis is

documented in I&T reports. Data, data logs, event logs and any other test output required for post test analysis is captured and stored under CM control.

3.3.3 Regression Testing

Regression testing is supplemental testing performed at any time upon any thread or build during I&T testing to ensure that existing software is not adversely affected by modified or new software. I&T members are responsible for planning, documenting, executing and reporting all regression testing. Automated test tools are used when practical, for regression testing by the I&T organization. This ensures that regression tests duplicate initial test procedures.

For Release A the following changes may result in regression testing:

- software changes
- hardware changes
- operational enhancements
- new versions delivered after the unit level testing

The I&T organization is responsible for reporting any discrepancies encountered during segment regression testing. Discrepancies resulting from any other level of testing which results in modifications at the unit level, will be regression tested by the I&T organization.

This page intentionally left blank.

Figure 3.2-1. Release A Build/Thread Diagram

Available in Hardcopy Only

Hold page for 11x17 figure.

4. CSMS Release A Test Procedures

The following sections identify the test procedures for conducting the Communications and Systems Segment Integration and System Test for Release A. CSMS is responsible for providing the software to support basic system and network communications (both locally and remotely), network and system security, system monitoring (to include the capability to locate, isolate, correct fault conditions), configuration management, and history logging/reporting of system activities.

First, threads are identified. Threads are the aggregation of unit tested components (CSCIs, CSUs, COTS software). Each thread demonstrates a CSMS function. Builds are the integration of threads and are identified after each series of threads which make up a build. Test cases and procedures are identified for each thread and build. The primary objective of each test case is to demonstrate and verify the capabilities of each function as stated in Level 4 requirements. All CSMS thread and build test cases will be conducted at the Landover ECS Development Facility (EDF).

4.1 Basic Communications Environment Tests

The following threads and builds verify the set of requirements related to internetworking and DCE security authentication:

- Internetworking Thread (TC001)
- DCE Security Authentication Thread (TC002)

4.1.1 Internetworking Thread (TC001)

This thread tests the internetworking infrastructure for Release A. Internetworking consists of the physical, data link, network, and transport layers of the OSI Reference Model. The tests will be conducted in the EDF on facilities emulating Release A DAAC LANs as closely as possible. The purpose of these tests is to ensure that the ISS is capable of providing the LAN connectivity between GSFC DAAC, LaRC DAAC, EDC DAAC, SMC, EOC, and SDPS/CSMS components. The initial configuration of the EDF is verified in this thread; later builds verify the complete communications infrastructure and Release A interfaces.

4.1.1.1 Test Case 1: Release A Initial Infrastructure Test (TC001.001)

The initial infrastructure test verifies the internetworking subsystem (ISS) services for TCP/IP over Ethernet and FDDI. Also the connectionless protocol, UDP, service must be provided in transport layer of OSI. The purpose of this test is to ensure that the ISS is capable of providing connectivity via the internet (through electronic mail) and NOLAN in a simulated environment.

Test Configuration:

Hardware: DCS Server, ISS components, LAN Analysis Equipment

Software: Emulated Release A mini-DAAC configuration, NOLAN emulation of protocols, NSI and ESN V0 connections

Data: Sample data files

Tools: HpOpenView, network analyzer

Test Input:

Connectivity will be verified by monitoring network functionality with HpOpenView, and by inspecting the hardware configurations and performing basic file transfer capabilities using E-Mail

Test Output:

Comparison of test data sent and received, and HpOpenView displays and reports.

Success Criteria:

This test is considered successful if data is correctly transported through the network with no faults logged or alarms triggererd. Also, HpOpenView diagnostics indicate no errors with network functionality.

Test Procedures:

Test Case ID: TC001.001	
Test Name: Internerworking Services	
Test Steps:	Comments:
1. Logon to the workstation	
2. Perform a FTP and run a telnet to any remote host **telnet** <host>; password **ftp** <host> username\ password **bye**	Facilitates the TCP/IP protocol suite
3. Utilize the '**netstat** command -**s** option (protocol statistics) - **a** option (all network interfaces)	
4. **cd /etc/host**	Examines the ASCII DB of all host #'s, names, and aliases of the other machines on the network
5. Initialize the network analyzer. - generate network traffic - filter/capture frames - display results in summary, detail, and hexadecimal	
6. Verify protocol usage over the internet	
7. Initialize HP OpenView **cd /usr/OV/bin** **ovw**	Currently configured on MSSE4HP workstation

8. View/report on network functionality and performance	
9. Inspect network activity, network configuration, and device configuration with the monitoring windows	
10. Display or print reports	
11. Initialize network trouble shooting commands from OS. **/etc/ping** *hostname [packet_size} {-n num. of packets}* **lanscan** - displays hw path, MAC address, LU number, state of hw and interface, device filename, and encapsulations methods of the local LAN **arp** *hostname* **-a** (all IP addresses to LLA mappings) **-d** (deletes an entry) **-s** (creates an entry)	
12. Verify network connectivity	Diagnose network connectivity > IP/TCP/UDP/SNMP Verifies: C-ISS-02000 C-ISS-02020 C-ISS-02030

4.1.1.2 Test Case 2: FDDI Switch Filtering Test (TC001.002)

This test case verifies the internetworking filtering of packets. The purpose of this test is to ensure that the ISS service is capable of filtering packets based on the port/socket of the transport layer protocol and/or the source/destination addresses of the network layer protocol at any DAAC site.

Test Configuration:

Hardware: FDDI switch on Atlantec Power Hub 7000, Concentrators on Sys 2914and cables, Ethernet Hub (Cabletron MMAC and ESX-1320), Routers, 1 sun workstation and three HPUX workstations

Software: Router configuration

Data: Test packets for TCP and UDP transport protocols

Tools: none

Test Input:

Inputs to this test consist of router configuration that establishes filtering conditions. Test packets with combination of port/socket identification and source/destination addresses will also be used as input.

Test Output:

Test output includes network management logs indicating the success or failure of packet transmission through the internetworking filtering.

Success Criteria:

This test is successful if invalid packets fail, allowed packets pass, and occurrences of failed packets are logged.

Test Procedures:

Test Case ID: TC001.002	
Test Name: Network Filtering Test	
Test Steps:	Comments:
1. From the first host [host A] telnet to the second host [host , logoff, and quit telnet **telnet**<host B> password **exit**	Test configuration will consist of four hosts connected by the port to the FDDI switch

2. From FDDI switch, setup a filter to block TCP/telnet. **Telnet 155.157.123.1**(fddi switch IP address) -Type '**ip**' to get the prompt for setting up filters -define template 1: **ip# ift add 1 b any 0.0.0.0 155.157.123.32 255.255.255.0 tcp = telnet or udp = talk** -define rule 2 to contain template 1: **ip# ifr add 2 1** -apply rule 2 to the port: **ip# ifa add 2 dst 2** - to show status: **ip# ifa s all -a** - to display ports and addresses: **ip # at**	(Hence: ift-if template, ifr- if rule, ifa- if application) FDDI SWITCH Prompt: >edf_fddi_switch: (0.0.0.0 - source; 155.157.123.32.255.255,255.0 - destination) **IMPORTANT:** No set address should be used while utilizing the udp protocol: ex. 155.157.123.0 This should set a filter block on host B. Example: dmge2hp Example Host A: csse3hp **Note**: For the udp protocol utilizing the talk, use SUN workstations just in case the HPs do not support the talk command!
3. Again, From the first host, telnet to the second host; [Talk to second host if using the udp protocol] telnet <host B>; password	This portion satisfies requirement: C-ISS-02030. No connection should occur since the **telnet** commands are blocked. No established connection should occur for the **talk command at this point for the udp protocol** Verifies a set filter based on TCP - C-ISS-02010
4. From the first host, FTP to the second host. A successful logon, logoff, and quit FTP **ftp** <host b>; username and password; **bye**	The second host responds and indicate that only TCP/Telnet commands are blocked

5. From the FDDI switch, remove TCP/Telnet filter and setup a filter to block a destination address for the second host -remove the current filter: **ip# ifa del all dst** **ip# ifr del 2 all** **ip# ift del all** -define template 2: **ip# ift add 2 b any 0.0.0.0 155.157.123.32**[HOST B] -define rule 1: **ip# ifr add 1 2** -apply rule 1 to port 1: **ip# ifa add 2 dst 1**	
6. From the first host, ping to the second host **telnet** <host B> **ping** **exit**	(this also satisfies requirement: C-ISS-02050) The second host does not respond because destination address for the second host is blocked Verifies the filter based upon the destination network layer (IP) - partial C-ISS-02040
7. From the first host, ping to a third host[host C] **telnet** <host B>; password **telnet** <host C.>; password **ping** **exit** < host C> **exit** < host B>	The third host responds showing that only the destination address for the second host is blocked

8. From the FDDI switch, remove the destination address filter and setup a filter to block the source address for the first host -remove the current filter: **ip# ifa del all dst** **ip# ifr del 1 all** **ip# ift del all** - define template 3: **ip# ift add 3 b 155.157.84.29[**HOST A] **255.255.255.255 any 0.0.0.0** -define rule 4 **ip# ifr add 4 3** -apply rule 4 to the port: **ip# ifa add 1 src 4**	Example: (155.157.84.29 is host A's ip address)
9. From the first host, ping the second host **telnet** <host B> **ping** **exit** <host b>	The second host does not respond because the source address for the first host is blocked Verifies the filter based upon the source network layer (IP) - partial C-ISS-02040
10. From the fourth host, ping to the second host	The second host responds because the source address for the fourth host is **NOT** blocked
11. From the FDDI switch, remove all filters: **ip# ifa del all src** **ip# ifa del all dst** **ip# ifr del 4 all** **ip# ift del all**	
12. Repeat steps 1 - 12 using alternate hosts/workstation or new ones	
13. Verify FDDI switch performance throughoutthe test	

4.1.1.3 Test Case 3: LAN Connectivity and OSI services Test (TC001.003)

This test case verifies that the CSMS provides communications and networking services at the EDC DAAC. ISS must provide LAN connectivity and four services: physical, data, network, and transport between CSMS components at EDC DAAC. External interfaces of the EDC DAAC are Landsat-7, V0 System, SMC, ASTER, and MODIS SCFs.

Test Configuration:

Hardware: FDDI switch on Atlantec Power Hub 7000, Ethernet Concentrators on Sys 2914 and cables, Ethernet Hub (Cabletron MMAC and ESX-1320), EBNet Router, Network interface cards, SYBASE Server, Science Processor, AIT Workstation, X-terminals, Printers for AIT and MSS,MSS Server, INGEST Server, and RAID

Software: ISS protocols in first four layers of OSI, especially the routing protocol: RIP of the network layer of OSI; Sybase, BuildersXcessory, SNMP agent, C, C++, DCE Client software

Data: EDC DAAC LAN topology, V0 DAAC LAN connections (helpful, but not necessary), NSI and L0 science data (helpful, but not necessary)

Tools: Network Analyzer

Test Input:

1. The EBNet WAN will serve as the interface between EDC, the DAACs, and NSI.
2. Traffic between ECS and NSI will flow via EDC ISOLAN. Internet communications for early interface testing to the EDC DAAC.
3. The EBNet is responsible for transporting spacecraft command, control, and L0 science data on a 24 hour, 7 days a week period using the real-time and science subnetworks.

Test Output:

Verification of V0 LAN connections on EBNet site diagram is shown. The DAAC interface to EBNet will be via single router with two interfaces to the DAAC. One interface must carry L0 science data for INGEST, and the interface will carry DAAC-to DAAC processing traffic as well as user traffic to and from NSI.

Success Criteria:

This test is successful if the network establish a direct connect to the FDDI switch.

Also, established connection and valid transmission of data utilizing the ISS components must be maintained.

Test Procedures:

Test Case ID: TC001.003	
Test Name: LAN connectivity and OS services	
Test Steps:	Comments:
	FDDI initialization from HTSC personnel.
1. Bring up the OpenView application.	
2. Verify the EBNet router is connected to the V0 WAN.	A FDDI switch check for V0 connections at DAAC sites Verifies: C-HRD-36065
3. Run telnet to associating host or IP address. **Telnet** <ip address or hostname>; password	Verifies: C-ISS-01256
4. Initiate a "**netstat** command" to verify the state of the current host.	

5. Remote login or utilize the NFS to authenticate hosts status.	
6. To find out which protocols are on your host, perform: **cat /etc/protocols**	
7. Use the "**ping**" command in the OS to monitor network performance and connections.	Information is displayed siting the source and destination addresses for test packets. Default packet size is 64 bytes.
8. Start the network analyzer. - FDDI Analyzer - Options: Default - Capture/Filter - Display results- - Save session or exit	
9. For ISS components, use the network analyzer to check the ISS protocols in the layers of the OSI Model	

4.1.1.4 Test Case 4: Network Layer Service Test (TC001.004)

This test case verifies that the internetworking subsystem utilizes the ICMP, IP,and RIP routing protocols in the network layer of OSI. The IP protocol provides datagram delivery, fragmentation and reassembly, and header data validation through checksums. The ICMP protocol provides a set of control messages operating as clients of the IP and this protocol is very useful for network management. The RIP protocol is the widely used routing protocol throughoout the ECS project for DAAC LANs, and its responsibility is to ensure data exchange, database maintenance and database access. The network service class relieves the transport service class of all routing and relay operations associated with network connection. As such, the network layer provides transparent transfer of data between transport entities.

Test Configuration:

Hardware: FDDI switch on Atlantec Power Hub 7000, Concentrators on Sys 2914and cables, Ethernet Hub (Cabletron MMAC and ESX-1320), Routers, Network interface cards, ARP server

Software: Network topoloy, LANconnections, ISS components

Data: network traffic, test packetized data

Tools: Network Analyzer

Test Input:

Data packets are sent to the LAN Analysis equipment.

Test Output:

Comparison of the source and destination addresses. Verify the ISS protocol connections and network traffic load using the network analyzer.

Success Criteria:

This test is successful if ISS network layer service supports protocol analysis and network traffic across the FDDI.

Test Procedures:

Test Case ID: TC001.004	
Test Name: Network Layer Service test	
Test Steps:	Comments:
1. Run the telnet to any selected host address. **Telnet** <to the FDDI switch>	Verifies C-ISS-02050 and C-ISS-02060.
2. Verify through the network management section, success or failures of source and destination.	
3. Initiate the network analyzer	
4. Designate a mode of operation to capture frames of data.	(IP over Ethernet)
5. Begin capturing/filtering	
6. Prepare various samples and diagnose the results from layer to layer.	
7. Examine the network layer of OSI to check the network addressing and routing problems if any	
8. Display protocol usage over FDDI.	Verifies C-ISS-02530
9. Generate network traffic and display captured frames.	
10. View the details for the destination class and the source stations	
11. Verify database access, maintenance, and exhange utilizing the routing protocol (RIP) within the network analyzer.	This may be performed by personnel permitted to validate the DB.
12. The subnet expert analyzer layer should provide information on all communicating subnetworks and the global layer should cover all other aspects such as multicasting and LAN overloads	

4.1.1.5 Test Case 5: LAN Transparent Portability Test (TC001.005)

This purpose of this test is to ensure that the LAN can support transparent portability across heterogeneous site LAN architecture. The management agent software component will be used to monitor and/or control managed objects distributed across heterogeneous platforms. Printing functions process FDDI-to-Ethernet.

Test Configuration:

Hardware: FDDI switch on Atlantec Power Hub 7000, Concentrators on Sys 2914 and cables, Ethernet Hub (Cabletron MMAC and ESX-1320) NOLAN/EBNet Router, INGEST Data Server, client hosts,working storage (RAID), workstations, printers

Software: Management agents, Clearcase, C,C++, ISS protocols,

Data: sample data files

Tools: None

Test Input:

Portability will be confirmed using the printing options within the LAN architecture.

Test Output:

A module, data file, or statement of conditions is printed from the user workstation. Remote login to perform printing functions at various locations. Documentation(s) is printed and examined for verification.

Success Criteria:

This test is successful if the print manager is able to perform print functions at a local or remote host.

Test Case ID: TC001.005	
Test Name: LAN Transparent Portability Test	
Test Steps:	Comments:
1. Using the operating system, perform a print locally and remotely. **lpr -P** < printer name> <filename>	Verifies: C-HRD-39000 (relates to ISS hardware)
2. Repeat step 1 using different terminals. (i.e SUN, HP, SGI, DEC, IBM)	Remote login

4.1.1.6 Test Case 6 Release 1 V0 WAN Test (TC001.006)

This Test Case verifies that there is an existing V0 WAN interface to both GSFC and LaRC DAACs for IR-1 connectivity. Both GSFC and LaRC DAACS will use existing V0 connections and external connections to the EBNet router.

Test Configuration:

Hardware: FDDI switch on Atlantec Power Hub 7000, FDDI Concentrators on Sys2914 and cables, Ethernet Hub (Cabletron MMAC and ESX-1320),EBNet router, NOLAN router for LaRC configuration, Network interfacecards, gateway, MSS and CSS Server and workstations, Ingest Servers, Planning DBMS Servers and an adjacent workstation, Queuing Server,AIT Server and workstation, Science Processors and X-terminals, Document Server, DD Server, FSMS Server, APC Server, DM DBMS Servers and workstations, an ops workstation, Repeater, and RAIDs

Software: TCP/IP and RIP protocols, NSI and V0 connections

Data: GSFC DAAC LAN and LaRC DAAC LAN topology , L0 and processing data

Tools: HP OpenView, Network performance tool or Network analyzer

Test Input:

1. For interoperability between V0 and V1, a gateway is set for connectivity.
2. The EBNet WAN will serve as the interface between EDC, the DAACs, and NSI.
3. The V0 network will be directly connected to the FDDI switch at both sites: GSFC and LaRC
4. ECS hosts supporting the IR-1 mission at EDC will have addresses assigned from the existing V0 address space.
5. The GSFC DAAC will have connections to the existing V0 network and to EBNet and the LaRC will utilize the same connections with an additional connection to the NOLAN router. Reuse of existing V0 connections is acceptable
6. Both GSFC and LaRC DAAC connections will be via separate FDDI interfaces on a single router. One interface is connected to the FDDI switch which will route data to the proper subsystems. This interface to the EBNet router carries both DAAC -to- DAAC processing flows as well as user traffic to and from NSI. The second interface does not directly connect to the FDDI switch, and a connection to the INGEST is performed to facilitate interface testing for L0 data. GSFC DAAC utilizes AM-1 MODIS L0 data, while the LaRC DAAC carries interface testing utilizing AM-1 CERES L0 data.
7. The EBNet is responsible for transporting spacecraft command, control, and science data on a 24 hour, 7 days a week period using the real-time and science subnetworks.

Test Output:

Site diagrams can depict network configuration. Verification of V0 connections on EBNet site diagram is shown. The EBNet utilizes two networks: real-time and science. The real-time network generates mission critical data related to health and safety of on orbit space systems and raw science telemetry as well as pre-launch and launch support. The science network (ESN) transports data collected from spacecraft instruments and other processed science data. Also, transmission of L0 data should describe an interface between the DAACs for IR-1 interface testing.

Success Criteria:

This test is considered successful if V0 WAN serves an interface between GSFC and LaRC DAAC for IR-1 connectivity. Established connection and valid transmission of data utilizing the

ISS components. The transmission of data should be kept online or stored during interface testing.

Test Case ID: TC001.006	
Test Name: V0 WAN Test	
Test Steps:	Comments:
1. Have HTSC verify the FDDI switch is up and running.	
2. Start the network analyzer.	
3. Verify the EBNet router is connected to the V0 WAN.	Verifies: C-ISS-01000, C-ISS-01010 C-ISS-01080, C-ISS-01090, C-HRD-36065
4. Check the FDDI switch for V0 connections at DAAC sites	
5. Use the "**ping**" command in the OS, to monitor network performance and connections	
6. For ISS components, use the network analyzer to check the ISS protocols in the layers of the OSI Model	

4.1.1.7 Test Case 7: EOC Infrastructure Connectivity Test (TC001.007)

This Test Case verifies that the GSFC EOC has EBNet connectivity. The EBNet WAN will serve as the interface between EDC, the DAACs, and NSI.

Test Configuration:

Hardware: Operational and Support FDDI, Concentrators on Sys 2914 and cables, Ethernet Hub (Cabletron MMAC and ESX-1320), EOC Router, Primary and Secondary Ecom Router, Network interface cards, RTS Server, DS Server, Data Archive Server, user workstations, RAID

Software: TCP/IP protocols, FTP and Kerberos software, NSI and V0 connections

Data: EBNet GSFC Site Design for TRMM Support, test packets,

Tools: OpenView, Network analyzer

Test Input:

Input from TC001.005 for GSFC DAAC and LaRC DAAC configuration. For GSFC EOC, the data server, data archives, and real-time servers must be directly connected to the FDDI. All hosts are connected to the operational and support FDDI. The EOC interfaces with the GSFC DAAC for exchange of instrument operations plans and schedules for the AM-1 spacecraft and mission information. The EBNet organization provides the IP addresses for network nodes and workstations. It should be noted that EOC is an element of the FOS Segment which implements the spacecraft analysis functions in support of the ECS FOS Subsystem (In Release B).

Test Output:

The OS provides the user with network status. The FDDI switch displays V0 connections at DAAC sites and EOC.

Success Criteria:

This test is successful if there are established network connections at GSFC EOC. The FDDI switch maintains host(s) to host(s) connections with no performance imbalances. The network analyzer monitors the network layer of OSI, and there are limited to none occurrences of addressing and routing problems.

Test Case ID: TC001.007	
Test Name: EOC Infrastructure Connectivity Test	
Test Steps:	Comments:
1. FDDI floppy boots up the software on the LAN.	
2. Start the network analyzer	
3. Verify the EBNet router is connected to the V0 WAN.	
4. Check the FDDI switch for EBNet connections at EOC. Connections for GSFC DAAC and LaRC DAAC need verification.	(Building 32 at Goddard). Verifies C-ISS-01195.
5. Use the "**ping**" command in the OS to monitor network performance and connections.	
6. Utilize the network analyzer to capture data, perform network filtering, and monitor ISS protocols within the layers of OSI.	
7. Demonstrate a data transfer using the kerberized file transfer protocol: **kftp**. Several attempts must be performed. Login to distributed computing environment **kinit** < username>; password **klist** **kftp** < host>; **bye** **kdestroy**	

4.1.1.8 Test Case 8: LLC and FDDI IEEE Standards Test (TC001.008)

This Test Case verifies the implementation of the physical and data link services of ISS compatible with the IEEE 802.2 (LLC) and the ANSI X3T9.5 (FDDI) standards.

Test Configuration:

Hardware: FDDI switch on Atlantec Power Hub 7000, Concentrators on Sys 2914 and cables, Ethernet Hub (Cabletron MMAC and ESX-1320), bridges,gateway, Sun workstation

Software: network configuration

Data: DAAC LAN designs

Tools: Network Analyzer

Test Input:

Review the DAAC LAN designs

Test Output:

Verify the DAAC LAN design has put in place the specific usage of the physical standards. The data link layer of the OSI-RM must provide connectivity and non-connectivity communications for the establishment, maintenance, and release of data link connections among the network entities.

Success Criteria:

This test is successful if the ISS standards are ISO compliant and the standards are compatible with logical link control and FDDI standards. The data link standard provides error-free transmission of frames of data.

Test Case ID: TC001.008	
Test Name: LLC and FDDI IEEE Standards Test	
Test Steps:	Comments:
1. Visually verify implementation of the DAAC LAN design.	
2. Review config. file containing the present configuration: hardware and software	
3. Initialze the network analyzer.	Obtain ISO compliant documentation. Verifies C-HRD-32000 (relates to ISS)
4. Set filters and begin capturing utilizing: Ethernet and FDDI analyzer.	
5. Display results. Verify error-free transmission of data.	
6. Submit to INGEST for verification	

4.1.1.9 Test Case 9: LAN Analysis Equipment Test (TC001.009)

The purpose of this test is verify the LAN Analysis Equipment can provide protocol analysis within the transport layer for all ISS and interconnection protocols to MANs.

Test Configuration:

Hardware: LAN Analysis Equipment, FDDI switch on Atlantec Power Hub 7000, Concentrators on Sys 2914 and cables, Ethernet Hub (Cabletron MMAC and ESX-1320), EBNet Router, Network interface cards, Ingest Data Server, Sun workstations, RAID

Software: TCP and UDP protocols, NSI and EBnet V0 connections, C, C++, Sybase V10, Sparcworks, OODCE 1.5.1, and DCE 1.0.3

Data: Network traffic, sample data files

Tools: Network expert analyzer, OpenView

Test Input:

1. EBnet connects to NOAA using existing V0 connections.
2. Establish hardware connection, then issue data packets across the network using TCP.
3. Transfer blocks of data to any two workstations in EDF using transport protocols
4. Utilize the LAN equipment to send data over the network within the transport layer

Test Output:

1. Established V0 connections
2. Error-free receipt of packetized blocks of data.
3. The TCP protocol establishes a connection with a remote terminal. Guareend delivery of data between hosts.
4. The UDP provides a connectionless transport with the absolute delivery to the recipient has no bit errors, but the sender may not be ensured the data was delivered.

Success Criteria:

This test is successful if the LAN Analysis equipment is capable of protocol analysis within the transport layer for all ISS and interconnection protocols to MANs.

Test Case ID: TC001.009	
Test Name: LAN Analysis Equipment Test	
Test Steps:	Comments:
1. Check to see if the INGEST hardware has a network analyzer as part of its functionality.	INGEST personnel needed
2. Start the network analyzer using the INGEST server.	
3. Review the ISS protocols within expert layer mode.	
4. Verify ISS protocols and interconnection protocols to the MAN and WAN.	Verifies C-HRD-34000(relates to ISS)
5. Gather ECS diagrams to verify software and hardware configuration.	
6. Refer to the M&O personnel for verification	HTSC personnel needed

4.1.1.10 Test Case 10: INGEST Test Equipment Test (TC001.010)

The purpose of this test is to verify that the INGEST hardware LAN Analysis Equipment incorporates a Communications line monitor which has the capability to store and display up to 10,000 bytes of sent and received data at the rates of 10Mbits/sec to 100 Mbits/sec. Also, the

Communications line monitor must support internetworking protocols and interconnectivity within ECS. This Test Case also verifies that the LAN analysis equipment includes Digital VOM/multi-meters and a network analyzer.

Test Configuration:

Hardware: LAN Analysis Equipment, Communication Line Monitor, FDDI switch onAtlantec Power Hub 7000, FDDI Concentrators on Sys 2914 and cables, Ethernet Hub (Cabletron MMAC and ESX-1320), EBNet Routers, RAID, L0 archive repository

Software: ISS protocols, C, C++, Sybase V10, Sparcworks, OODCE 1.5.1, and DCE 1.0.3

Data: L0 data, test packets

Tools: Network analyzer

Test Input:

1 .FDDI configuration of network devices and software are interconnected, and the communication lines shall suport ISS protocols utilizing the FDDI switch.

2. LAN Analysis equipment provides protocol verification and connection.

3. Test packets are sent over the communication lines and verifiy receipt. Echo requests and provide route verification.

Test Output:

1. The session establishment sequence is captured so that the network analyzer recognizes the protocol. Connectivity is established with any remote system.

2. In the second level of the network analyzer, cable testing is performed, verified, and displayed.

3. A comparison between sent data and received data is performed.

4. A diagnosis report on the network functionality was generated from OpenView

Success Criteria:

This test will be successful if the data rates of 10MB/sec to 100 MB/sec through the communications lines over FDDI are maintaining interconnectivity and supporting the ISS protocols.

Test Case ID: TC001.010	
Test Name: INGEST Equipment Test	
Test Steps:	Comments:
1. Verify a LAN analyzer is located in the mini-DAAC.	
2. Verify communications lines are visible and set.	

3. Verify FDDI specifications as listed in the network analyzer operation's manual	
4. Using the network analyzer, verify ISS protocols used within ECS.	
5. Inspect ISS interconnection units used : bridges, routers, repeaters, gateways, and other products to finalized a unified networking system	HTSC personnel assistance Verifies C-HRD-34010 (relates to ISS components)

4.1.1.11 Test Case 11: EOC LAN Test (TC001.011)

The purpose of this test is to verify through real-time operations that the EOC LAN loop delay contribution does not exceed more than 500 msec. (goal of 250 msec) of the total ECS delay of 2.5 second for emergency RT commands. The forward and return links must have times within the total ECS delay period. The EOC Operational LAN must also be capable of peak traffic rate of 24Mbps on the backbone.

Test Configuration:

Hardware: EOC LAN network configuration, Operational and Support FDDIs,Concentrators on Sys 2914 and cables, Ethernet Hub (Cabletron MMAC and ESX-1320), Routers, Network interface cards, RTS Server, Data Servers and File Server, RAID, Data Archives, user workstations, unicast or multicast-ISTs

Software: ECL, C, C++, Clearcase, operating system, Sparcworks, IDL

Data: EOC configuration design, data files and command loads

Tools: HP OpenView, command control display(ECL)

Test Input:

1. Demonstrate the token ring timer: negotiate how long it takes to loop around per command sent to the LAN.
2. Calculate the raw data traffic estimates as provided by the FOS design team
3. Utilizing the FDDI switch to verify the loop delay contribution.
4. Generate real-time commands uplink to spacecraft via automatic ground script or via command controller in ECL.
5. Compare loop delay contribution per command load.

Test Output:

1. The EOC Operational LAN backbone utilizes the FDDI network, withstands network failure between LANs and continues operation using any available LAN. The backbone

also regulates filtering and network traffic are permitted accordingly. Peak traffic rate is 20MB/s for EOC Operational Network backbone.

2. A display derived from the FDDI switch verifies nominal loop delay rates.
3. Loop delay contribution has reached the goal of 250 msec of total ECS.
4. Receipt of real-time commands downlink to controller. Verify loop delay contributions on the LAN. Calculation of the delay time <= 2.5 msec. for real-time commands.

Success Criteria:

This test will be successful if the command controller provide authorization checks, validated the commands, and uplinks via EDOS. The loop delay contribution must not exceed more than 500 msec. of the total ECS delay of 2.5 second for real-time commands.

Test Case ID: TC001.011	
Test EOC LAN Test	
Test Steps:	Comments:
Logon to workstation.	
Bring up OpenView.(MSS Server)	
Initialize ECL.	
Access the command controller	
Enter start/stop times, orbit, station, and pass initialization	
Demonstrate commanding concurrently or independently using the command control display	
Verify receipt of valid commands and display the command status report	
Archive commands in the command buffer for re-transmissions	
Utilizing the token timer, monitor the data files pertaining to the loop delay contribution on the LAN.	
Gather data from the FDDI switch and display results. The loop delay contribution must have a range less than or equal to 500 milliseconds of the total ECS delay of 2.5 second for real-time commands.	
Exit OpenView	
Calculate delay times per real-time uplink/downlink command.	
Report results	

The FOS design team provides raw data traffic estimates for the AM-1 mission. . Data rates (actual data on LAN) = **raw data** + **communications overhead** Raw data converted to network traffic by adding ***overhead***. As data increases, data is multiplied by **factor of 7**. Review EOC total peak estimates chart in the FOS design specification.	Verifies: C-HRD-36000, C-HRD-36010
15. Halt command control and terminate the pass	

4.1.1.12 Test Case 12: Network Performance Test (TC001.012)

The purpose of this test is to verify the interoperability of the ISS LAN at GSFC and LaRC DAACs can support trice the R-A network traffic load estimates and the Operational LAN must be capable of supporting 230 network devices with peak data rates of up to 48 Mbps without redesign. Information is transmitted between ECS and NSI to enable network communications and network management.

Test Configuration:

Hardware: Operational and support FDDI , Concentrators on Sys 2914 and cables, Ethernet Hub (Cabletron MMAC and ESX-1320), NOLAN/EbNet Router (s), Working Storage, DSS Server, Science processor, RTS Server, Ingest Data Server, and SMP Server, RAID, workstations and x-terminals, L0 archive repository

Software: OS, , Clearcase, C, C++, SNMP protocol, TCP/IP protocol, and other ISS protocols from the network layer to the application layer

Data: Appendix A of ECS External Data Traffic Requirements or Communications Requirements for ECS, TRMM metadata, sample files

Tools: Network expert analyzer

Test Input:

Network loads are generated over the Ethernet. Data flows from DAAC-to-DAAC. Packets of data generated by TCP/IP. Optimum operational threshold levels are to be defined for each device and network facility.

Test Output:

Reports on the status of the LAN data flow estimate and speed. Uptime process to show what hosts are on the LAN and whether these hosts are up and running is generated and checked for relevancy.

Success Criteria:

This test will be successful if at least 230 devices with the expected traffic rate is in accordance to the communication requirements for ECS. Services and the load of traffic on the LAN must maintain operability and high performance.

Test Case ID: TC001.012	
Test Name: Network Performance Test	
Test Steps:	Comments:
Logon to the workstation	
Telnet to a host and perform file transfers	
Bring up the network analyzer.	
Demonstrate the expert mode for network analysis.	
Demonstrate network connectivity through the network layer of the OSI model.	
Set up interfaces for TRMM data to flow from GSFC and LaRC DAAC.	
Using the network analyzer, monitor and verify network statistics of DAAC data flow.	
Verify the number of network devices supported and percentage of network traffic utilization.	
Using the network analyzer, generate frames to load the network .	
The network layer routes data packets from the source to the destination.	
Check the OS for the network status for the selected host and internetworking hosts at site locations: GSFC and LaRC.	
Evaluate the network analyzer's report on the network utilization and load estimates	
Using the Communications Requirements document, provide a comparison for the network traffic load and performance. ECS real-time and non-real time data flows are included for traffic estimates from DAAC-to-DAAC in the data traffic requirements.	

Verify utilization of devices, responsiveness of devices, and congestion conditions.	
Gather data and print reports.	Verifies: C-HRD-36070, C-HRD-36090, C-HRD-36100

4.1.1.13 Test Case 13: LAN Expansion Test (TC001.013)

This purpose of this test to provide a faster, higher speed communication channel to enable an expansion to GByte networks including the ability to provide increased volume of data distribution and access. The SDPS subsystem also provides for the distribution of data electronically or on physical media. The current Release A configuration utilizes the SCSI communication channel. The migration path to Release B for the LANs inserts the HIPPI communication channel to the configuration.

Test Configuration:

Hardware: FDDI switch on Atlantec Power Hub 7000, Concentrators on Sys 2914 and cables, Ethernet Hub (Cabletron MMAC and ESX-1320), NOLAN/EBNet Router, INGEST Data Server, client hosts, working storage (RAID), DBMS repository, workstations, printers, planning server, FSMS server, APC server, Media Distribution server, and science processor

Software: Clearcase, C. C++, ISS protocols,

Data: Release A and Release configuration design, L0 data through SDPS

Tools: Openview, twork expert analyzer

Test Input:

The expansion to GByte networks for Release A will provide an increased volume of data distribution and access. The network configuration upgrade places the HIPPI communication channel parallel to the FDDI network. The HIPPI insertion can produce high performance point-to point channel connection between high-end systems.

Test Output:

The HIPPI communication channel provides an interface layer to support the Upper Layered protocols ranging from the network layer to the application layer. Using HIPPI produces very high TCP/IP performance on the network.

Success Criteria:

This test will be successful if the migration to the HIPPI insertion parallel to the FDDI network increases the volume of data distribution and access. Also, the network performance is sufficient and prone to limited to none internetworking failures.

Test Case ID: TC001.013	
Test Name: LAN Expansion Test	
Test Steps:	Comments:
1. Have the upgraded hardware been attached to the ISS components in the mini DAAC for Release B?	Verifies C-HRD-39005 (relates to ISS components)

4.1.1.14 Test Case 14: Release A Router Configuration Test (TC001.014)

The purpose of this test is to verify that a local DAAC or NMF has backups of all router configuration files contained on the boot-up floppy on the FDDI. Also, verify the maintenance and operations group can acquire an archive through the FSMS.

Test Configuration:

Hardware: FDDI switch on Atlantec Power Hub 7000, floppy/storage device on FDDI, Concentrators on Sys 2914 and cables, Ethernet Hub (Cabletron ,MMAC and ESX-1320), Routers, Network interface cards, HP Server, RAID

Software: HPUX 9.05 operating system, Management Services, C, C++, Clearcase

Data: Router configuration data, CM logs

Tools: ClearCase Configuration Management Tool

Test Input:

Verify the FDDI switch contains a storage device. The management and operations group may generate reports upon request.

Test Output:

The operating system boots up floppy on the FDDI. Router configuration files are viewed using telnet and verification through the management model.

Success Criteria:

This test is successful if high performance is held for the FDDI switch, the operating system substains failure, and no data is lost on the storage device.

Test Procedures:

Test Case ID: TC001.014	
Test Name: Router Configuration Test	
Test Steps:	Comments:
1. Verify FDDI switch is mounted on disk.	This task is performed by the HTSC personnel. Hence: Backups must be made prior to utilizing the disk. Verifies: C-ISS-04020
2. Access ASCII router configuration files.	
3. Data is stored and archived on a daily basis.	(HTSC personnel)
4. Acquire archives from HTSC (M&O) group for validation.	Approval of valid information must be set by NMF or a local mini-DAAC.
5. Check storage capacity of the floppy.	(M&O verification)

4.1.1.15 Test Case 15: SMC Function Test (TC001.015)

The purpose of this test is ensure the ISS services at the SMC at Goddard can support the SMC functions of gathering and disseminating system management information's availability requirement of .998 and MDT of less than twenty minutes during the staffed operation. External interfaces of SMC are EBnet, EOC, and TSDIS, and Release A DAACs. Each ISS LAN segment must have operational availability and sufficient MDT for ECS functions it supports. Due to the high availability required for SMC, the FDDI LAN must be implemented via physically wired rings as opposed to concentrators.

Test Configuration:

Hardware: Dual FDDI switches on Atlantec Power Hub 7000, six FDDI cables, four Ethernet cables, Ethernet-to-FDDI hub, Concentrators, APC Server, Routers, Network interface cards, Network management stations (NMS), Communications (CSS) and Management (MSS) Server, BB Server, three SUN Sparc(s), RAID, HP LaserJet Printer, HP 9000 workstations, DCE client, Sybase server, X-server

Software: C, C++, Clearcase, SNMP agent software,report generation, trouble ticker and fault performance management specifically on the MSS server, DCE name and security on CSS server

Data: Sybase DBMS files, logistics data management, toolkit information; Maintainability predictions report

Tools: Hp OpenView, Sybase applications, network analyzer

Test Input:

Management of the ECS operations have set up the interface for internetworking subsystem to perform transparent data transfers within the SMC to other site locations. Access Hp Openview to gather database files that contain information about SMC's managed objects such as servers, workstations, peripherals, BB, and FDDI concentrators. Utilize Openview to allow the management software to provide system monitoring and control. SMC's configuration must be maintained in the CSS and MSS servers.

Test Output:

Sybase generated reports and SNMP queries from SMC at Goddard Space Flight Center (GSFC). The SNMP agents will contain network information regarding the number of connections and the speed of transmission at specific locations. Established bandwidth to handle operational availability. The ILS and logistics data management reports system and segment mean down time for the SMC functions.

Success Criteria:

This test is considered successful if the system availability of 99.8% or above and the MDT of twenty minutes or less has been established using SMC functions. Also, the test will be successful if the internetworking services are provided at SMC located in building 32 at Goddard.

Test Procedures:

Test Case ID: TC001.015	
Test Name: SMC Function Test	
Test Steps:	Comments:
1. Bring up Openview, and at the host or through remote login, focus on some database parameters to be used to test ratios for availability and MDT	
2. Bring up the network analyzer.	
3. Using the network analyzer, monitor the network bandwidth utilization. Increase and/or decrease the bandwidth to obtain nominal rate for the SMC's operational availability.	
4. Evaluate the mean down time and operational availability to be .<=.998 in a range less than twenty minutes. Compare status to that of ILS and logistics management. Overhead factor in reason could be less than or equal to 74% communication line usage and the protocol overhead factor less than or equal to 25%.	The MTBM and MTTR are obtained from the COTs vendor.
5. Verify MTBM (mean time between maintenance) Formula for operational availability: **A = MTBM + MDT**	

6. Formula for MDT: **MDT = MTTR (*mean time to repair*) + 4 hours of estimated logistics delaytime**	**Hence**: the MTBM consists of MTBPM (mean time between preventive maintenance) and MTBF (mean time between failures). The time interval given by the COTS vendor is the MTBPM.
7. Remote login and perform file transfers to an associating DAAC.	
8. Repeat the 3rd and 4th step. Track the results.	Verifies: C-ISS-04155

4.1.1.16 Test Case 16: SNMP agents verification and MIB II Variable Validation (TC001.016)

The purpose of this Test Case is to query MIB II which is Host MIB, and Network device MIB information from the Host Resource Agent on Solaris and HP-UX machines by using the HP OpenView MIB browser. The MSS Monitor/Control Service is not complete as of phase 1, the MSS Management Agent Service only responds to requests for managed objects MIB II attributes, and provides agent for network devices.

Test Configuration:

Hardware: MSS Server, Managed resources: host(i.e. Workstation) and application(s),Inter networking subsystem components (i.e. LAN Network resources)

Software: HpOpenView, MSS-MACI components, Peer Network SNMP master agent on the host, MsAgAgent-COTS, MsAgEncps-COTS, MIB II Agent

Data: MIB definition

Tools: HpOpenView

Test Input:

Inputs to this Test Case include HpOpenView commands to examine the contents of the host MIB II.

Test Output:

Outputs include the MIB II contents displayed on HpOpenView screen.

Success Criteria:

This test is successful if the ISS components can verify via SNMP agents, and MIB II values are displayed via HpOpenView match those specified in the MIB definition documentation.

Test Case ID: TC001.016	
Test Name: SNMP agents verification and MIB II Variable Validation	
Test Steps:	Comments:
1. Verify the SUN snmp process is running by executing the following command: **ps -ef | grep snmpd**	Configuring and starting the Host Resource Agent on Solaris. Verifies : C-ISS-02110
2. If the snmpd process is listed, kill it by executing the following command: **kill -9** processid #	Note: This requires root access.
3. Enter the PEER master agent directory: **cd /usr/peer/agent/build/src/wrk**	Starting the PEER master agent.
4. Check the PEER master agent configuration file to reflect the manager(s) that may access the master agent. The current included file "MsAgAgent.cfg" will allow one machine, cyclops **,** to query the master agent.	
5. Start the master agent in the background: **./MsAgAgent MsAgAgent.cfg filename &**	Note: You must be root. Any filename is acceptable.
6. Enter the PEER encapsulator directory: **cd /usr/peer/subagts/encaps/src/wrk**	Starting the PEER Encapsulator.
7. Start the encapsulator in the background: **./MsAgEncps -c MsAgEncps.cfg -s 1999 &**	
8. Enter into the Solaris Host Resource MIB directory: **cd /opt/SUNWconn/snm/agents**	Starting the Solaris Host Resource agent.
9. Start the Solaris host resource agent: **/opt/SUNWconn/snm/agents/snmpd -c /etc/opt/SUNWconn/snm/snmpd.conf -p 1161 &**	Verify that the MSS Management Agent service provides agent for network devices -MSS-36070. Verifies: C-ISS-02110
10. Enter the Host name in the box entitled "Name or IP Address".	
11. Enter "public" in the box entitled "Community Name".	
12. Go to the MIB branch entitled: .iso.org.dod.internet.mgmt	
13. All sub-branches should now be available to query.	
14. Query all groups in MIB II (e.g. System, Interfaces, Address Translation (AT), IP, ICMP, TCP, UDP, EGP groups)	Verify ability to respond to requests for managed object MIB attributes - MSS-36020.
15. Enter the IP address in the box entitled "Name or IP Address".	
16. Repeat steps 11 through 14 as above.	Verify ability to respond to requests for managed object MIB attributes - MSS-36020.
17. Verify the HP-UX snmp process is running by executing the following command: **ps -ef | grep snmpd**	Configuring and starting the Host Resource Agent on HP-UX. Verifies: C-ISS-02110

18. If the snmpd process is listed, kill it by executing the following command: **kill -9** processid#	Note: This requires root access.
19. Check and make sure that the snmp port number is 1161 in the **/etc/services** file.	
20. Start the HP-UX host resource agent: **/etc/snmpd &**	Verify that the MSS Management Agent service provides agent for network devices -MSS-36070.
21. Start the master agent in the background: **./MsAgAgent MsAgAgent.cfg filename &**	Starting the PEER master agent. Note: You must be root. Same filename as 5.
22. Enter the PEER encapsulator directory: **cd /usr/peer/subagts/encaps/src/wrk**	Starting the PEER Encapsulator.
23. Start the encapsulator in the background: **./MsAgEncps -c MsAgEncps.cfg -s 1999 &**	
24. Query the Host Resource Agent Information by starting the HP OpenView MIB browser: **/usr/OV/bin/xnmbrowser**	It is assumed that the agent is running on the machine that you wish to query, and that HP OpenView has been properly configured to query the HP and Solaris Host Resource Agents.
25. Repeat steps 10 through 16 as above for HP-UX.	
26. Modify the SUBTREES of the MsAgEncps.cfg (e.g., change SUBTREES 1.3.6.1.2.1.2 to 1.3.6.1.2.1).	Forcing known information into the host resource.
27. Modify the snmpd.conf file (change any of the sysdeer, syscontact, syslocation...)	
28. If the snmpd process is listed, kill it by excuting the following command: **kill -9** processid #	
29. Restart the snmpd process: **/opt/SUNWconn/snm/agents/snmpd -c /etc/opt/SUNWconn/snm/snmpd.conf -p 1161 &**	
30. Check the HP OpenView MIB browser.	Make sure that the modified information is shown on the HP OpenView MIB browser.

4.1.2 DCE Security Authentication Thread Test (TC002)

This thread verifies the ability to perform DCE security authentication. Authentication is the process of verifying the validity of a principal. It is performed when users login to the ECS domain. The authentication mechanism is based on two parameters: principal identities and secret keys, or passwords. Initially, the server uses the userid and password of the user who invoked it as its principal identity and key.

4.1.2.1 Test Case 1: Invalid Login (TC002.001)

This test case demonstrates that the CSS Security Service will reject any invalid login attempts due to invalid usernames and invalid passwords.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: rgy_edit utility

Data: Invalid User login name and password

Tools: None

Test Input:

Attempts are made to login to an account with both invalid username and invalid password.

Test Output:

Messages indicating the success or failure of login attempts are displayed.

Success Criteria:

Users with invalid usernames and/or invalid passwords are successfully rejected in their attempts to log into the system.

Test Procedures:

Test Case ID: TC002.001	
Test Name: Invalid Login	
Test Steps:	Comments
1. Login attempt with invalid DCE Userid	
2. Login to DCE client.	
3. Attempt DCE login with **invalid** username. **"dce_login <username>"**	
4. Enter any password.	
5. Verify login attempt rejected.	ERROR: User Id failure. Registry object not found (dce/sec) ***C-CSS-21000***
6. Login attempt with invalid DCE Password	
7. Login to DCE client.	
8. DCE login with **valid** username. **"dce_login <username>"**	
9. Enter **invalid** password.	
10. Verify login attempt rejected.	ERROR: Password Validation Failure. Invalid Password (dce/sec) ***C-CSS-21000***

4.1.2.2 Test Case 2: Create New DCE User and Account (TC002.002)

This test verifies the ability to create new DCE user accounts and privileges in the security registry.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: rgy_edit utility

Data: User login name and password

Tools: None

Test Input:

The DCE cell administrator creates a new user account and a password using the rgy_edit utility.

Also, a new DCE Organization and new Group are required before creating user account. (see TC002.005 and TC002.006)

Test Output:

A user account is created in the security registry.

Success Criteria:

A new user account is successfully created using rgy_edit.

Test Procedures:

Test Case ID: TC002.002	
Test Name: Create New DCE User and Account	
Test Steps:	Comments
Create New DCE User	Editor for Cell Registry.
1. Enter: rgy_edit	
2. rgy_edit => **do principle**	
3. rgy_edit => **add**	Enter a Userid name
4. Add Principle => Enter Name : <***username>***	Automatically assigned
5. Enter UNIX number: ***Press Return***	
6. Enter Full Name in quotes : "**user full name**"	unlimited
7. Enter object creation quota : **0**	New userid created
8. Add Principle => Enter Name : ***Press Return***	
Create New DCE Account	domain account
9. rgy_edit=> **do account**	
10. rgy_edit=> **add**	
11. Add Account=> Enter Account id: **<username>**	
12. Enter account group: **<groupname>**	
13. Enter account organization: **<organization name>**	
14. Enter password: **<user password>**	
15. Retype password: **<user password>**	
16. Enter your password: **<cell_admin password>**	
17. Enter misc info: () <**organization full name> (<organization name>)**	
18. Enter home directory: (/) **/home**	
19. Enter shell: () /bin/csh	Press Return 14 times.
20. ***Press Return*** to accept all other defaults	
21. rgy_edit=> **view <username>**	
22. Verify account information.	***C-CSS-21020***
23. rgy_edit=> exit	

4.1.2.3 Test Case 3: Modify a DCE User Password (TC002.003)

This test verifies the ability to modify a DCE user password.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: rgy_edit utility

Data: User login name

Test Input:

The DCE cell administrator modifies a user password using the rgy_edit utility.

Test Output:

Modified user password allows user to log into the system.

Success Criteria:

Modified user password allows user to successfully log into the system, while the old password is rejected.

Test Procedures:

Test Case ID: TC002.003	
Test Name: Modify a DCE User Password	
Test Steps:	Comments
1. Enter: rgy_edit	Editor for Cell Registry.
2. rgy_edit => **do account**	
3. rgy_edit => **change -mp <cell_admin password> -pw <new password>**	Modify user password.
4. Change Account=> Enter Account id: **<username>**	
5. Enter account group: **<groupname>**	
6. Enter account organization: **<organization name>**	
7. rgy_edit=> exit	
8. Perform DCE login. **"dce_login <username>"**	
9. Enter DCE **<oldpassword>**	
10. Verify old password **rejected**.	
11. Perform DCE login. **"dce_login <username>"**	
12. Enter DCE **<newpassword>**	Enter new password created.
13. Verify newpassword **login successful**.	***C-CSS-21020***

4.1.2.4 Test Case 4: Delete DCE User Login (TC002.004)

This test verifies the ability to delete a DCE user login.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: rgy_edit utility

Data: User login name and password

Tools: None

Test Input:

The DCE cell administrator deletes a user login using the rgy_edit utility.

Test Output:

A user login is deleted from the security registry.

Success Criteria:

A user login is successfully deleted using rgy_edit.

Test Procedures:

Test Case ID: TC002.004	
Test Name: Delete DCE User Login	
Test Steps:	Comments
1. Enter: rgy_edit	Editor for Cell Registry.
2. rgy_edit => **do principle**	Domain username.
3. rgy_edit => **view <username>**	View user listing.
4. Verify username exist.	
5. rgy_edit => **delete <username>**	
6. Please confirm delete of name "**<username>**" [y/n] ? **y**	
7. rgy_edit => **view <username>**	
8. Cannot retrieve entry for **<username> - Entry not found**	Verify username is deleted from user list.
9. rgy_edit => **exit**	
10. Enter: **kdestroy**	Kill kerberos ticket
11. Enter: **exit**	exit DCE
12. Attempt DCE login. **"dce_login <username>"**	
13. Verify login is **rejected** successfully.	***C-CSS-21020***

4.1.2.5 Test Case 5: Create New DCE Organization (TC002.005)

This test verifies the ability to create a new DCE Organization.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: rgy_edit utility

Data: User login name and password

Tools: None

Test Input:

The DCE cell administrator creates a DCE Organization using the rgy_edit utility.

Test Output:

A DCE organization is created in the security registry.

Success Criteria:

A new DCE organization is successfully created using rgy_edit.

Test Procedures:

Test Case ID: TC002.005	
Test Name: Create New DCE Organization	
Test Steps:	Comments
1. Enter: rgy_edit	Editor for Cell Registry.
2. rgy_edit => **do org**	domain **organization**
3. rgy_edit => **add**	
4. Add Org => Enter Name: **<organization name>**	
5. Enter UNIX number: ***Press Return***	
6. Enter Full Name in quotes : "**organization full name**"	
7. Add Org => Enter Name : ***Press Return***	
8. rgy_edit=> exit	***C-CSS-21020***

4.1.2.6 Test Case 6: Create New DCE Group (TC002.006)

This test verifies the ability to create a new DCE group.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: rgy_edit utility

Data: User login name and password

Tools: None

Test Input:

The DCE cell administrator creates a new DCE group using the rgy_edit utility.

Test Output:

A DCE group account is created in the security registry.

Success Criteria:

A new group account is successfully created using rgy_edit.

Test Procedures:

Test Case ID: TC002.006	
Test Name: Create New DCE Group	
Test Steps:	Comments
1. Enter: rgy_edit	Editor for Cell Registry.
2. rgy_edit => **do group**	domain **group**
3. rgy_edit => **view**	view current groups
4. rgy_edit => **add**	
5. Add Group => Enter Name: **<groupname>**	
6. Enter UNIX number: ***Press Return***	
7. Enter Full Name in quotes : "**group full name**"	
8. Include group on PROJLIST [y/n] ? **y**	
9. Add Group => Enter Name : ***Press Return***	
10. rgy_edit=> **view**	***C-CSS-21030***

4.1.2.7 Test Case 7: Modify DCE Group Account (TC002.007)

This test verifies the ability to modify a new DCE group account.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: rgy_edit utility

Data: User login name and password

Tools: None

Test Input:

The DCE cell administrator modifies a DCE group account using the rgy_edit utility.

Test Output:

A DCE group account is modified in the security registry.

Success Criteria:

A DCE group account is successfully modified using rgy_edit.

Test Procedures:

Test Case ID: TC002.007	
Test Name: Modify DCE Group Account	
Test Steps:	Comments
1. Enter: rgy_edit	
2. rgy_edit => **do account**	
3. rgy_edit => **change -ng <newgroupname>**	Modify user from old group to new group
4. Change Account=> Enter account id: **<username>**	
5. Enter account group: **<oldgroupname>**	

6. Enter account organization: **<organization name>**	
7. rgy_edit => **view <username>**	
8. Verify username is member of new group	**<username> [<newgroupname> < >] :*:**
9. rgy_edit => do group	
10. rgy_edit => view **<newgroupname> -m**	
11. Again verify username is member of new group	Alternative check ***C-CSS-21030***

4.1.2.8 Test Case 8: Add a User to DCE Group (TC002.008)

This test verifies the ability to add a user to the DCE group.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: rgy_edit utility

Data: User login name and password

Tools: None

Test Input:

The DCE cell administrator will add a user to a DCE group using the rgy_edit utility.

Test Output:

A user is added to a DCE group in the security registry.

Success Criteria:

A user is successfully added to a DCE group using rgy_edit.

Test Procedures:

Test Case ID: TC002.008	
Test Name: Add a User to DCE Group	
Test Steps:	Comments
1. Enter: rgy_edit	Editor for Cell Registry.
2. rgy_edit => **do group**	
3. rgy_edit => **view**	view current groups
4. rgy_edit => **member <groupname> -a <username>**	Adding a user to a group
5. rgy_edit => **view <groupname> -m**	view additional **member (-m)** to **<groupname>**
6. Verify username added to groupname	
7. rgy_edit => exit	***C-CSS-21020***

4.1.2.9 Test Case 9: Delete User from a DCE Group (TC002.009)

This test verifies the ability to delete a user from the DCE group.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: rgy_edit utility

Data: User login name and password

Tools: None

Test Input:

The DCE cell administrator delete a user from a DCE group using the rgy_edit utility.

Test Output:

A user is deleted from a DCE group in the security registry.

Success Criteria:

A user is successfully deleted from a DCE group using rgy_edit.

Test Procedures:

Test Case ID: TC002.009	
Test Name: Delete User from a DCE Group	
Test Steps:	Comments
1. Enter: rgy_edit	Editor for Cell Registry.
2. rgy_edit => **do group**	
3. rgy_edit => **member <groupname> -r <username>**	**-r** removes user from group.
4. Warning: any accounts for (**<groupname> <username>**) will be deleted.	
5. Please confirm delete of name **<groupname>** [y/n] ? **y**	
6. rgy_edit => **view <groupname> -m**	
7. Verify **<username>** member is **Deleted**	***C-CSS-21020***

4.1.2.10 Test Case 10: Delete a DCE Group (TC002.010)

This test verifies the ability to delete a DCE group.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: rgy_edit utility

Data: User login name and password

Tools: None

Test Input:

The DCE cell administrator delete a DCE group using the rgy_edit utility.

Test Output:

A DCE group is deleted from the security registry.

Success Criteria:

A DCE group is successfully deleted using rgy_edit.

Test Procedures:

Test Case ID: TC002.010	
Test Name: Delete a DCE Group	
Test Steps:	Comments
1. Enter: rgy_edit	Editor for Cell Registry.
2. rgy_edit => **do group**	
3. rgy_edit => **delete <groupname>**	Deletes group.
4. Please confirm delete of name **<groupname>** [y/n] ? **y**	
5. rgy_edit => **view**	View all groups.
6. Verify group is **Deleted** from group list.	***C-CSS-21030***

4.1.2.11 Test Case 11: Password Encryption (TC002.011)

This test verifies the ability to protect user passwords by not transmitting them in clear text across networks. By default, all user passwords are encrypted in a security keytab file when the password is created for the users account.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: rgy_edit utility

Data: User login name and password

Tools: None

Test Input:

rgy_edit utility

Reference Documentation:

OSF DCE Application Development Guide

OSF DCE Administration Guide - Core Components

Test Output:

The user passwords contained in the keytab file are encrypted.

Success Criteria:

User passwords are encrypted within the security keytab file.

Test Procedures

Test Case ID: TC002.011	
Test Name: Password Encryption	
Test Steps:	Comments
	See DCE security login process documented in the OSF DCE Application Development Guide chp. 42 section 2. Also, see OSF DCE Administration Guide chp. 20 section 6 Table 20-1 Account Information Summary.
1. rgy_edit => <enter>	
2. rgy_edit => **ktadd -p principle -pw <password> -f EcSeServerKeyFile**	Create password for **newuserx** to be added to the keytab file.
3. rgy_edit => **ktlist -p principle -f <keytab name>**	list keytab
4. rgy_edit => quit	
5. Enter: **more EcSeServerKeyFile**	Verify Keytab entry password is encrypted. Note: the format is: <dce_cell><username><password encrypted> **C-CSS-21010**

4.1.2.12 Test Case 12: Login Expiration/Refresh (TC002.012)

This test verifies that the CSS Security Service provides an API to limit the time after which a login context expires. Also, the ability to refresh login contexts before they expire is tested via the server.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: rgy_edit utility

Data: User login name and password, Login expiration time specified.

Tools: None

Test Input:

A DCE authenticated user creates a new user account and a password using the rgy_edit utility. The login contexts for this user are allowed to expire in a predetermined period of time. The test is repeated, however, this time the login contexts are refreshed before the expiration time limit is reached.

Test Output:

Messages indicating the login expiration or refresh occurs.

Success Criteria:

Login contexts expire after the predetermined expiration time is reached, and not before. Logins do not expire if they are refreshed for an extended amount of time by the administrator but do expire at the extended expiration time setting successfully.

Test Procedures:

Test Case ID: TC002.003	
Test Name: Change Expiration Time for Login Context	
Test Steps:	Comments
1. Enter: rgy_edit	
2. rgy_edit => **do account**	
3. rgy_edit => **view <username> -f**	
4. Verify **current** Account expiration date: 19yy/mm/dd.hh:mm	Verify the current expiration date/time.
5. rgy_edit => **change -x yy/mm/dd.hh:mm**	Change user account expiration date/time.
6. Change Account=> Enter account id: **<username>**	
7. Enter account group: **<groupname>**	
8. Enter account organization: **<organization name>**	
9. rgy_edit => **view <username> -f**	
10. Verify the Account expiration date/time is changed.	
11. rgy_edit => **exit**	
12. Perform DCE login. **"dce_login <username>"**	Login **before** the expiration date/time expires.
13. Enter DCE **password**	
14. Enter: klist	
15. Verify login is successful.	
16. Wait until expiration date/time has expired.	
17. Perform DCE login. **"dce_login <username>"**	Login **after** the expiration date/time expires.
18. Enter DCE **password**	
19. Sorry. Password Validation Failure. - account not valid for login	Verify login is successfully **rejected**. ***C-CSS-21040***
20. Enter: **kdestroy**	Destroy kerberos ticket.
21. Enter: **exit**	exit DCE
Test Step: Refresh for Login Context Expiration	Comments
1. Enter: rgy_edit	
2. rgy_edit => **do account**	
3. rgy_edit => **view <username> -f**	
4. Verify **current** Account expiration date: 19yy/mm/dd.hh:mm	Verify the current expiration date/time.
5. rgy_edit => **change -x yy/mm/dd.hh:mm**	Changing user account expiration date/time.
6. Change Account=> Enter account id: **<username>**	

7. Enter account group: **<groupname>**	
8. Enter account organization: **<organization name>**	
9. rgy_edit => **view <username> -f**	
10. Verify the Account expiration date/time is changed.	
11. rgy_edit => **exit**	
12. Perform DCE login. **"dce_login <username>"**	Login **before** the expiration date/time expires.
13. Enter DCE **password**	
14. Verify login successful.	***C-CSS-21050***
15. Enter: klist	
16. Enter: kdestroy	Destroy ticket.
17. Enter: klist	
18. Verify ticket is destroyed.	
19. Enter: exit	

4.1.2.13 Test Case 13: Server Key Processing (TC002.013)

This test verifies the ability of the CSS Security Service to provide APIs to process server keys as follows:

- Provide a unique session key for each client session.
- Accept server keys associated with services interactively at the startup of a service.
- Store server keys associated with servers to a disk file.
- Retrieve server keys associated with services from a disk file at startup time to authenticatethe service.
- Change the identity of an application process through server keys.

This test also verifies that an API is provided to challenge the client/server to authenticate itself at the following three levels: connect level, request level, and packet level.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h, rgy_edit utility, DCEMemPassword (DCE COTS)

Data: User login name and password

Tools: None

Test Input:

The DCE cell administrator creates a new user account and password using the rgy_edit utility.

Test Output:

The correct assignment, retrieval, and storage of server keys is verified, as well as the ability to change the identity of an application through server keys.

Success Criteria:

The correct assignment, retrieval, and storage of server keys is successfully demonstrated, as well as the ability to change the identity of an application through server keys.

Test Procedures:

Test Case ID: TC002.013	
Test Name: Server Key Storage to Disk File	
Test Steps:	Comments
1. dce_login <cell_admin> <cell_admin password>	
2. rgy_edit => <enter>	
3. rgy_edit => **ktadd -p <principle name> -pw <password> -f <keytadd filename>**	Add server key to cell keytab file on disk. ***C-CSS-21070***
4. rgy_edit => **ktlist -p <principle name> -f <keytab name>**	list keytab
5. Verify unique client session key created.	***C-CSS-21005***
	This new server principle and password is verified in the following **Automatic Key Retrieval at Server Startup** procedure.
Test Steps: Automatic Key Retrieval at Server Startup	Comments
1. Create unique username and password in the keytabfile.	Created in previous **Server Key Storage to Disk File** procedure.
2. cd **/usr/testa/IT/tools/CSS/SEC**	
3. Start application server. Enter: **server &**	**server** is a script file consisting of the commands to run the server executable along with the **client** script file.
4. Verify that server is listening.	
5. Start application client. Enter **client**.	Client will login into server. **client** is a script file consisting of the commands to run the client executable.
6. Verify client has successfully logged on to the Server.	***C-CSS-21080***
7. Enter: **Kill -9 xxxx**	
8. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	
Test Steps: Change Application Identity with Server Keys	Comments
1. **cp client_chgid client**	Changes client application identity to **newuserx**
2. dce_login <cell_admin> <cell_admin password>	
3. rgy_edit => <enter>	
4. rgy_edit => **ktadd -p newuserx -pw <password> -f <keytadd filename>**	Create password for **newuserx** to be added to the keytab file.
5. rgy_edit => **ktlist -p newuserx -f <keytab name>**	list keytab
6. cd **/usr/testa/IT/tools/CSS/SEC**	

7. Start application server. Enter: **server &**	**server** is a script file consisting of the commands to run the server executable along with the **client** script file.
8. Verify that server is listening.	
9. Start application client. Enter **client**.	Client will login into server. **client** is a script file consisting of the commands to run the client executable.
10. Verify that the changed client has successfully logged on to the Server.	***C-CSS-21090***
11. Enter: **Kill -9 xxxx**	Kill server process
12. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	
Test Steps: Interactive Key Retrieval at Server Startup	Comments
1. Login to <platform> workstation in **edfcell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
2. cleartool setview <username>	Setup user clearcase view
3. cd **/ecs/formal/CSS/REL_A**	
4. Enter: **source .buildrc**	
5. cd **/ecs/formal/CSS/REL_A/DOF/src/SEC/test**	
6. Enter: **clearmake -C gnu clean**	Cleanup old **header** and **object** files
7. Enter: **cp ServerMain.cxx.pwd ServerMain.cxx**	Set **server** software to accept
8. Enter: **clearmake -C gnu**	Compile software from clearcase
9. Copy **ServerMain.cxx** to your home directory.	
10. Login to <platform> workstation in **relacell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
11. cd **/usr/testa/IT/tools/CSS/SEC**	Location for test driver execution.
12. **ftp <workstation name> .hitc.com** ex. ftp casper.hitc.com	ftp to edfcell.
13. > **cd**	change directory to your home directory
14. > **ls**	List files in directory
15. > **bin**	binary mode
16. > **prompt**	disable prompting
17. > **mget S***	Transfers **Sleeper** files to the directory
18. Wait until all files are transferred	
19. > **quit**	exit ftp
20. cd **/usr/testa/IT/tools/CSS/SEC**	
21. Start application server. Enter: **server &**	**server** is a script file consisting of the commands to run the server executable along with the **client** script file.
22. Enter server key password at prompt.	
23 Verify server key accepted.	Interactive server key at startup.
24. Start application client. Enter **client**.	**client** is a script file consisting of the commands to run the client executable.
25. Verify client has successfully logged on to the Server.	***C-CSS-21060***
26. Enter: **Kill -9 xxxx**	Kill server process
27. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	

4.1.2.14 Test Case 14: Authn Preferences (Connect Level) (TC002.014)

This test verifies that an API is provided to challenge the client/server to authenticate itself at the connect level.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: Server and Client authentication information and protection levels

Tools: OODCE 1.0.3, RogueWave Tools.h++ 6.1

Test Input:

Authentication of the client/server with the connect level protection (rpc_c_protect_level_connect).

Test Output:

Messages indicating the success or failure of the login attempts are displayed.

Success Criteria:

Client/server authentication is successfully performed at the connect level.

Test Procedures:

Test Case ID: TC002.014	
Test Name: Authentication Preferences (Connect Level)	
Test Steps:	Comments
1. Login to <platform> workstation in **edfcell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
2. cleartool setview <username>	Setup user clearcase view
3. cd **/ecs/formal/CSS/REL_A**	
4. Enter: **source .buildrc**	
5. cd **/ecs/formal/CSS/REL_A/DOF/src/SEC/test**	
6. Enter: **clearmake -C gnu clean**	Cleanup old **header** and **object** files
7. Enter: **cp ServerMain.cxx.conn ServerMain.cxx**	Set **server** software for RPC Connect Level Protection **(rpc_c_protect_level_connect)**
8. Enter: **clearmake -C gnu**	Compile software from clearcase
9. Enter: **cp SleeperClient.cxx.conn SleeperClient.cxx**	Set **client** software for RPC Connect Level Protection **(rpc_c_protect_level_connect)**
10. Enter: **clearmake -C gnu**	Compile software from clearcase

11. Copy **ServerMain.cxx** and **SleeperClient.cxx** to your home directory.	
12. Login to <platform> workstation in **relacell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
13. cd **/usr/testa/IT/tools/CSS/SEC**	Location for test driver execution.
14. **ftp <workstation name> .hitc.com** ex. ftp casper.hitc.com	ftp to edfcell.
15. > **cd**	change directory to your home directory
16. > **ls**	List files in directory
17. > **bin**	binary mode
18. > **prompt**	disable prompting
19. > **mget S***	Transfers **Sleeper** files to the directory
20. Wait until all files are transferred	
21. > **quit**	exit ftp
22. Perform DCE login. **"dce_login <username>"**	
23. Enter DCE **password**	
24. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory.
25. Start application server. Enter: **server_conn &**	
26. Verify that server is listening.	
27. Login to remote client.	Remote client login.
28. Perform DCE login. **"dce_login <username>"**	
29. Enter DCE **password**	
30. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory on client.
31. Start application client. Enter **client_conn**.	Client will login into server.
32. Verify RPC is processed.	**Remote sleep entered for object UUID *C-CSS-21100***
33. Enter: **Kill -9 xxxx**	xxxx -> PID for **Sleeper_Server**
34. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	

4.1.2.15 Test Case 15: Authn Preference (Request Level) (TC002.015)

This test verifies that an API is provided to challenge the client/server to authenticate itself at the request level.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: User login name and password

Tools: OODCE 1.0.3, RogueWave Tools.h++ 6.1

Test Input:

Authentication of the client/server at the request level.

Test Output:

Messages indicating the success or failure of the login attempts are displayed.

Success Criteria:

Client/server authentication is successfully performed at the request level.

Test Procedures:

Test Case ID: TC002.015	
Test Name: Authentication Preference (Request Level)	
Test Steps:	Comments
1. Login to <platform> workstation in **edfcell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
2. cleartool setview <username>	Setup user clearcase view
3. cd **/ecs/formal/CSS/REL_A**	
4. Enter: **source .buildrc**	
5. cd **/ecs/formal/CSS/REL_A/DOF/src/SEC/test**	
6. Enter: **clearmake -C gnu clean**	Cleanup old **header** and **object** files
7. Enter: **cp ServerMain.cxx.call ServerMain.cxx**	Set **server** software for RPC Connect Level Protection **(rpc_c_protect_level_call)** Note: The application server software will be set to generate UDP data packets.
8. Enter: **clearmake -C gnu**	Compile software from clearcase
9. Enter: **cp SleeperClient.cxx.call SleeperClient.cxx**	Set **client** software for RPC Connect Level Protection **(rpc_c_protect_level_call)**
10. Enter: **clearmake -C gnu**	Compile software from clearcase
11. Copy **ServerMain.cxx** and **SleeperClient.cxx** to your home directory.	
12. Login to <platform> workstation in **relacell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
13. cd **/usr/testa/IT/tools/CSS/SEC**	Location for test driver execution.
14. **ftp <workstation name> .hitc.com** ex. ftp casper.hitc.com	ftp to edfcell.
15. > **cd**	change directory to your home directory
16. > **ls**	List files in directory
17. > **bin**	binary mode
18. > **prompt**	disable prompting
19. > **mget S***	Transfers **Sleeper** files to the directory
20. Wait until all files are transferred	
21. > **quit**	exit ftp
22. Perform DCE login. **"dce_login <username>"**	
23. Enter DCE **password**	
24. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory.
25. Start application server. Enter: **server_call &**	
26. Verify that server is listening.	
27. Login to remote client.	Remote client login.
28. Perform DCE login. **"dce_login <username>"**	

29. Enter DCE **password**	
30. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory on client.
31. Start application client. Enter **client_call**.	Client will login into server.
32. Verify RPC is processed.	**Remote sleep entered for object UUID *C-CSS-21100***
33. Enter: **Kill -9 xxxx**	xxxx -> PID for **Sleeper_Server**
34 Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	

4.1.2.16 Test Case 16: Authn Preference (Packet Level) (TC002.016)

This test verifies that an API is provided to challenge the client/server to authenticate itself at the packet level.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: User login name and password

Tools: OODCE 1.0.3, RogueWave Tools.h++ 6.1

Test Input:

Authentication of the client/server at the packet level.

Test Output:

Messages indicating the success or failure of the login attempts are displayed.

Success Criteria:

Client/server authentication is successfully performed at the packet level.

Test Procedures:

Test Case ID: TC002.016	
Test Name: Authentication Preference (Packet Level)	
Test Steps:	Comments
1. Login to <platform> workstation in **edfcell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
2. cleartool setview <username>	Setup user clearcase view
3. cd **/ecs/formal/CSS/REL_A**	
4. Enter: **source .buildrc**	
5. cd **/ecs/formal/CSS/REL_A/DOF/src/SEC/test**	
6. Enter: **clearmake -C gnu clean**	Cleanup old **header** and **object** files

7. Enter: **cp ServerMain.cxx.pkt ServerMain.cxx**	Set **server** software for RPC Connect Level Protection **(rpc_c_protect_level_pkt)**
8. Enter: **clearmake -C gnu**	Compile software from clearcase
9. Enter: **cp SleeperClient.cxx.pkt SleeperClient.cxx**	Set **client** software for RPC Connect Level Protection **(rpc_c_protect_level_pkt)**
10. Enter: **clearmake -C gnu**	Compile software from clearcase
11. Copy **ServerMain.cxx** and **SleeperClient.cxx** to your home directory.	
12. Login to <platform> workstation in **relacell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
13. cd **/usr/testa/IT/tools/CSS/SEC**	Location for test driver execution.
14. **ftp <workstation name> .hitc.com** ex. ftp casper.hitc.com	ftp to edfcell.
15. > **cd**	change directory to your home directory
16. > **ls**	List files in directory
17. > **bin**	binary mode
18. > **prompt**	disable prompting
19. > **mget S***	Transfers **Sleeper** files to the directory
20. Wait until all files are transferred	
21. > **quit**	exit ftp
22. Start application server. Enter: server &	
23. Verify that server is listening.	
24. Login to DCE client.	Remote client login.
25. Perform DCE login. **"dce_login <username>"**	
26. Enter DCE **password**	
27. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory on client.
28. Start application client. Enter **client**.	Client will login into server.
29. Verify RPC is processed.	***C-CSS-21100***
30. Enter: **Kill -9 xxxx**	xxxx -> PID for **Sleeper_Server**
31. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	

4.1.2.17 Test Case 17: Data Encryption (Connect Level) (TC002.017)

This test verifies the ability of the CSS Security Service to provide APIs to encrypt and decrypt data at the **connect level**, along with Data integrity which is guaranteed by using checksums. The Data Encryption Standard (DES) is supported to perform the encryption and decryption of data.

This testcase satisfies the stated connect level requirements ***C-CSS-21170, C-CSS-21180, C-CSS-21190, C-CSS-21200*** by using both the Inspection and Test, process methods.

The *Inspection* portion verifies that the DCE COTS product meets the stated requirements in regard to the functionality of the DCE Encrypt, Decrypt, and Integrity checking mechanisms. DCE COTS uses the Data Encryption Standard (DES) as stated in their documentation, see Reference OSF DCE Application Development Guide Section 42.1.5 pg. 42-5. Also, the data integrity checking using the checksum mechanism is a sub-function of the encryption mechanism used by the DES as documented in the OSF Application Environment Specification/Distributed

Computing - Security Volume chp. 3 section 3.3. The DCE COTS references describe the functionality provided by DCE , which also meets the CSS Security Service requirements as stated.

The *Test* portion verifies that the connect level protection preference is set in the client application. Once set, the DCE runtime automatically invokes the appropriate security code to authenticate the server. To provide mutual authentication (that is, allow the server to check that the client is as claimed) and to ensure that the server is willing to meet the client preferences for protection, a reference monitor must be used.

OODCE defines an abstract base class for a reference monitor object named **DCERefMon**. DCERefMon is an abstract class that provides an interface to reference monitor protection level functionality.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: User DCE login name and password

Tools: OODCE 1.0.3, RogueWave Tools.h++ 6.1

Test Input:

Client API set for connect level protection.

Test Output:

Test verifies that the client application protection level preference is set properly before it gets passed to the server. From that point on, DCE COTS completes the transaction.

Success Criteria:

This test is deemed successful when the DCE connect level protection preference gets set properly in the client application.

Test Procedures

Test Case ID: TC002.017	
Test Name: Encryption/Decryption (Connect Level)	
Test Steps:	Comments
1. Login to <platform> workstation in **edfcell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
2. cleartool setview <username>	Setup user clearcase view
3. cd **/ecs/formal/CSS/REL_A**	
4. Enter: **source .buildrc**	

5. cd **/ecs/formal/CSS/REL_A/DOF/src/SEC/test**	
6. Enter: **clearmake -C gnu clean**	Cleanup old **header** and **object** files
7. Enter: **cp ServerMain.cxx.conn ServerMain.cxx**	Set **server** software for RPC Connect Level Protection **(rpc_c_protect_level_connect)**
8. Enter: **clearmake -C gnu**	Compile software from clearcase
9. Enter: **cp SleeperClient.cxx.conn SleeperClient.cxx**	Set **client** software for RPC Connect Level Protection **(rpc_c_protect_level_connect)**
10. Enter: **clearmake -C gnu**	Compile software from clearcase
11. Copy **ServerMain.cxx** and **SleeperClient.cxx** to your home directory.	
12. Login to <platform> workstation in **relacell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
13. cd **/usr/testa/IT/tools/CSS/SEC**	Location for test driver execution.
14. **ftp <workstation name> .hitc.com** ex. ftp casper.hitc.com	ftp to edfcell.
15. > **cd**	change directory to your home directory
16. > **ls**	List files in directory
17. > **bin**	binary mode
18. > **prompt**	disable prompting
19. > **mget S***	Transfers **Sleeper** files to the directory
20. Wait until all files are transferred	
21. > **quit**	exit ftp
22. Start application server. Enter: server &	
23. Verify that server is listening.	
24. Login to DCE client.	Remote client login.
25. Perform DCE login. **"dce_login <username>"**	
26. Enter DCE **password**	
27. cd **/usr/testa/IT/tools/CSS/SEC**	Location for test driver execution.
28. Start application server. Enter: <**server>&**	
29. Start application client. Enter: <**client>**	Client will login into server.
30. Verify connect level protection ID number displayed.	This verifies that the client application protection level parameter is properly set.
31. Verify RPC is processed.	
32. Enter: **Kill -9 xxxx**	xxxx -> PID for <**Sleeper_Server>**
33. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	

4.1.2.18 Test Case 18: Data Encryption (Request Level) (TC002.018)

This test verifies the ability of the CSS Security Service to provide APIs to encrypt and decrypt data at the **request level**, along with Data integrity which is guaranteed by using checksums. The Data Encryption Standard (DES) is supported to perform the encryption and decryption of data.

This testcase satisfies the stated **request** level requirements ***C-CSS-21170, C-CSS-21180, C-CSS-21190, C-CSS-21200*** by using both the Inspection and Test, process methods.

The *Inspection* portion verifies that the DCE COTS product meets the stated requirements in regard to the functionality of the DCE Encrypt, Decrypt, and Integrity checking mechanisms. DCE COTS uses the Data Encryption Standard (DES) as stated in their documentation, see Reference OSF DCE Application Development Guide Section 42.1.5 pg. 42-5. Also, the data integrity checking using the checksum mechanism is a sub-function of the encryption mechanism used by the DES as documented in the OSF Application Environment Specification/Distributed Computing - Security Volume chp. 3 section 3.3. The DCE COTS references describe the functionality provided by DCE , which also meets the CSS Security Service requirements as stated.

The *Test* portion verifies that the request level protection preference is set in the client application. Once set, the DCE runtime automatically invokes the appropriate security code to authenticate the server. To provide mutual authentication (that is, allow the server to check that the client is as claimed) and to ensure that the server is willing to meet the client preferences for protection, a reference monitor must be used.

OODCE defines an abstract base class for a reference monitor object named **DCERefMon**. DCERefMon is an abstract class that provides an interface to reference monitor protection level functionality.

Test Configuration:

Hardware:	SunOS 5.4, HP 9.05
Software:	Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h
Data:	User DCE login name and password
Tools:	OODCE 1.0.3, RogueWave Tools.h++ 6.1

Test Input:

Client API set for request level protection.

Test Output:

Test verifies that the client application protection level preference is set properly before it gets passed to the server. From that point on, DCE COTS completes the transaction.

Success Criteria:

This test is deemed successful when the DCE request level protection preference gets set properly in the client application.

Test Procedures

Test Case ID: TC002.017	
Test Name: Encryption/Decryption (Request Level)	
Test Steps:	Comments
1. Login to <platform> workstation in **edfcell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
2. cleartool setview <username>	Setup user clearcase view
3. cd **/ecs/formal/CSS/REL_A**	
4. Enter: **source .buildrc**	
5. cd **/ecs/formal/CSS/REL_A/DOF/src/SEC/test**	
6. Enter: **clearmake -C gnu clean**	Cleanup old **header** and **object** files
7. Enter: **cp ServerMain.cxx.conn ServerMain.cxx**	Set **server** software for RPC Request Level Protection **(rpc_c_protect_level_request)**
8. Enter: **clearmake -C gnu**	Compile software from clearcase
9. Enter: **cp SleeperClient.cxx.conn SleeperClient.cxx**	Set **client** software for RPC Request Level Protection **(rpc_c_protect_level_request)**
10. Enter: **clearmake -C gnu**	Compile software from clearcase
11. Copy **ServerMain.cxx** and **SleeperClient.cxx** to your home directory.	
12. Login to <platform> workstation in **relacell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
13. cd **/usr/testa/IT/tools/CSS/SEC**	Location for test driver execution.
14. **ftp <workstation name> .hitc.com** ex. ftp casper.hitc.com	ftp to edfcell.
15. > **cd**	change directory to your home directory
16. > **ls**	List files in directory
17. > **bin**	binary mode
18. > **prompt**	disable prompting
19. > **mget S***	Transfers **Sleeper** files to the directory
20. Wait until all files are transferred	
21. > **quit**	exit ftp
22. Start application server. Enter: server &	
23. Verify that server is listening.	
24. Login to DCE client.	Remote client login.
25. Perform DCE login. **"dce_login <username>"**	
26. Enter DCE **password**	
27. cd **/usr/testa/IT/tools/CSS/SEC**	Location for test driver execution.
28. Start application server. Enter: <**server>&**	
29. Start application client. Enter: <**client>**	Client will login into server.
30. Verify request level protection ID number displayed.	This verifies that the client application protection level parameter is properly set.
31. Verify RPC is processed.	
32. Enter: **Kill -9 xxxx**	xxxx -> PID for <**Sleeper_Server>**
33. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	

4.1.2.19 Test Case 19: Data Encryption (Packet Level) (TC002.019)

This test verifies the ability of the CSS Security Service to provide APIs to encrypt and decrypt data at the **packet level**, along with Data integrity which is guaranteed by using checksums. The Data Encryption Standard (DES) is supported to perform the encryption and decryption of data.

This testcase satisfies the stated **packet** level requirements ***C-CSS-21170, C-CSS-21180, C-CSS-21190, C-CSS-21200*** by using both the Inspection and Test, process methods.

The *Inspection* portion verifies that the DCE COTS product meets the stated requirements in regard to the functionality of the DCE Encrypt, Decrypt, and Integrity checking mechanisms. DCE COTS uses the Data Encryption Standard (DES) as stated in their documentation, see Reference OSF DCE Application Development Guide Section 42.1.5 pg. 42-5. Also, the data integrity checking using the checksum mechanism is a sub-function of the encryption mechanism used by the DES as documented in the OSF Application Environment Specification/Distributed Computing - Security Volume chp. 3 section 3.3. The DCE COTS references describe the functionality provided by DCE , which also meets the CSS Security Service requirements as stated.

The *Test* portion verifies that the **packet** level protection preference is set in the client application. Once set, the DCE runtime automatically invokes the appropriate security code to authenticate the server. To provide mutual authentication (that is, allow the server to check that the client is as claimed) and to ensure that the server is willing to meet the client preferences for protection, a reference monitor must be used.

OODCE defines an abstract base class for a reference monitor object named **DCERefMon**. DCERefMon is an abstract class that provides an interface to reference monitor protection level functionality.

<u>Test Configuration:</u>

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: User DCE login name and password

Tools: OODCE 1.0.3, RogueWave Tools.h++ 6.1

<u>Test Input:</u>

Client API set for **packet** level protection.

Test Output:

Test verifies that the client application protection level preference is set properly before it gets passed to the server. From that point on, DCE COTS completes the transaction.

Success Criteria:

This test is deemed successful when the DCE **packet** level protection preference gets set properly in the client application.

Test Procedures

Test Case ID: TC002.019	
Test Name: Encryption/Decryption (Packet Level)	
Test Steps:	Comments
1. Login to <platform> workstation in **edfcell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
2. cleartool setview <username>	Setup user clearcase view
3. cd **/ecs/formal/CSS/REL_A**	
4. Enter: **source .buildrc**	
5. cd **/ecs/formal/CSS/REL_A/DOF/src/SEC/test**	
6. Enter: **clearmake -C gnu clean**	Cleanup old **header** and **object** files
7. Enter: **cp ServerMain.cxx.conn ServerMain.cxx**	Set **server** software for RPC Packet Level Protection **(rpc_c_protect_level_packet)**
8. Enter: **clearmake -C gnu**	Compile software from clearcase
9. Enter: **cp SleeperClient.cxx.conn SleeperClient.cxx**	Set **client** software for RPC Packet Level Protection **(rpc_c_protect_level_packet)**
10. Enter: **clearmake -C gnu**	Compile software from clearcase
11. Copy **ServerMain.cxx** and **SleeperClient.cxx** to your home directory.	
12. Login to <platform> workstation in **relacell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
13. cd **/usr/testa/IT/tools/CSS/SEC**	Location for test driver execution.
14. **ftp <workstation name> .hitc.com** ex. ftp casper.hitc.com	ftp to edfcell.
15. > **cd**	change directory to your home directory
16. > **ls**	List files in directory
17. > **bin**	binary mode
18. > **prompt**	disable prompting
19. > **mget S***	Transfers **Sleeper** files to the directory
20. Wait until all files are transferred	
21. > **quit**	exit ftp
22. Start application server. Enter: server &	
23. Verify that server is listening.	
24. Login to DCE client.	Remote client login.
25. Perform DCE login. **"dce_login <username>"**	
26. Enter DCE **password**	
27. cd **/usr/testa/IT/tools/CSS/SEC**	Location for test driver execution.
28. Start application server. Enter: <**server>&**	

29. Start application client. Enter: <**client>**	Client will login into server.
30. Verify packet level protection ID number displayed.	This verifies that the client application protection level parameter is properly set.
31. Verify RPC is processed.	
32. Enter: **Kill -9 xxxx**	xxxx -> PID for <**Sleeper_Server>**
33. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	

4.1.2.20 Test Case 20: DCE Shared-Secret Key (TC002.020)

The purpose of this test is to demonstrate that the DCE shared-secret key authentication mechanism is offered to clients by the server authentication protocols provided.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: User login name and password

Tools: OODCE 1.0.3, RogueWave Tools.h++ 6.1

Test Input:

Inputs to this test case include valid username and password for DCE client login and client request for DCE shared-secret key authentication method.

Test Output:

The expected results of this test case is a successful authentication.

Success Criteria

This test will be deemed successful when the user is logged on successfully.

Test Procedures:

Test Case ID: TC002.020	
Test Name: DCE Shared-Secret Key Authentication	
Test Steps:	Comments
1. Login to <platform> workstation in **edfcell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
2. cleartool setview <username>	Setup user clearcase view
3. cd **/ecs/formal/CSS/REL_A**	
4. Enter: **source .buildrc**	
5. cd **/ecs/formal/CSS/REL_A/DOF/src/SEC/test**	
6. Enter: **clearmake -C gnu clean**	Cleanup old **header** and **object** files

7. Enter: **cp ServerMain.cxx.authn_secret ServerMain.cxx**	Set **server** software for RPC authentication **(rpc_c_authn_dce_secret)**
8. Enter: **clearmake -C gnu**	Compile software from clearcase
9. Enter: **cp SleeperClient.cxx.authn_secret SleeperClient.cxx**	Set **client** software for RPC authentication **(rpc_c_authn_dce_secret)**
10. Enter: **clearmake -C gnu**	Compile software from clearcase
11. Copy **ServerMain.cxx** and **SleeperClient.cxx** to your home directory.	
12. Login to <platform> workstation in **relacell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
13. cd **/usr/testa/IT/tools/CSS/SEC**	Location for test driver execution.
14. **ftp <workstation name> .hitc.com** ex. ftp casper.hitc.com	ftp to edfcell.
15. > **cd**	change directory to your home directory
16. > **ls**	List files in directory
17. > **bin**	binary mode
18. > **prompt**	disable prompting
19. > **mget S***	Transfers **Sleeper** files to the directory
20. Wait until all files are transferred	
21. > **quit**	exit ftp
22. Start application server. Enter: server &	Server will autolog itself and start listening.
23. Verify that server is started.	
24. Start application client. Enter **client**.	Client request authentication into server.
25. Verify client successfully logs on.	***C-CSS-21110***
26. Enter: **Kill -9 xxxx**	xxxx -> PID for **sleeper_server**
27. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	

4.1.2.21 Test Case 21: Authentication Preference None (TC002.021)

The purpose of this test is to demonstrate that "no authentication" is offered to clients by the server authentication protocols provided.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: User login name and password

Tools: OODCE 1.0.3, RogueWave Tools.h++ 6.1

Test Input:

Inputs to this test case include valid username and password for DCE client login and client request for no authentication method.

Test Output:

The expected results of this test case is a successful login.

Success Criteria

This test will be deemed successful when the user is logged in.

Test Procedures:

Test Case ID: TC002.021	
Test Name: No Authentication (Authentication Preference)	
Test Steps:	Comments
1. Login to <platform> workstation in **edfcell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
2. cleartool setview <username>	Setup user clearcase view
3. cd **/ecs/formal/CSS/REL_A**	
4. Enter: **source .buildrc**	
5. cd **/ecs/formal/CSS/REL_A/DOF/src/SEC/test**	
6. Enter: **clearmake -C gnu clean**	Cleanup old **header** and **object** files
7. Enter: **cp ServerMain.cxx.authn_none ServerMain.cxx**	Set **server** software for RPC authentication **(rpc_c_authn_none)**
8. Enter: **clearmake -C gnu**	Compile software from clearcase
9. Enter: **cp SleeperClient.cxx.authn_none SleeperClient.cxx**	Set **client** software for RPC authentication **(rpc_c_authn_none)**
10. Enter: **clearmake -C gnu**	Compile software from clearcase
11. Copy **ServerMain.cxx** and **SleeperClient.cxx** to your home directory.	
12. Login to <platform> workstation in **relacell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
13. cd **/usr/testa/IT/tools/CSS/SEC**	Location for test driver execution.
14. **ftp <workstation name> .hitc.com** ex. ftp casper.hitc.com	ftp to edfcell.
15. > **cd**	change directory to your home directory
16. > **ls**	List files in directory
17. > **bin**	binary mode
18. > **prompt**	disable prompting
19. > **mget S***	Transfers **Sleeper** files to the directory
20. Wait until all files are transferred	
21. > **quit**	exit ftp
22. Start application server. Enter: server &	Server will autolog itself and start listening.
23. Verify that server is started.	
24. Start application client. Enter **client**.	Client will login into server.
25. Verify client successfully logs on.	***C-CSS-21110***
26. Enter: **Kill -9 xxxx**	xxxx -> PID for **sleeper_server**
27. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	

4.1.2.22 Test Case 22: Mismatch Authentication (TC002.022)

The purpose of this test is to demonstrate that when the server authentication method does not match the clients desired method, authentication is denied and notification is posted.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: User login name and password

Tools: OODCE 1.0.3, RogueWave Tools.h++ 6.1

Test Input:

Inputs to this test case include valid username and password for DCE client login and client desires no authentication, whereas the server is defined for DCE shared-secret key authentication.

Test Output:

The expected results of this test case is an unsuccessful authentication.

Success Criteria

This test will be deemed successful when the user is properly denied login due to mismatched preferences.

Test Procedures:

Test Case ID: TC002.022	
Test Name: Mismatched Authentication Preferences	
Test Steps:	Comments
1. Login to <platform> workstation in **edfcell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
2. cleartool setview <username>	Setup user clearcase view
3. cd **/ecs/formal/CSS/REL_A**	
4. Enter: **source .buildrc**	
5. cd **/ecs/formal/CSS/REL_A/DOF/src/SEC/test**	
6. Enter: **clearmake -C gnu clean**	Cleanup old **header** and **object** files
7. Enter: **cp ServerMain.cxx.authn_secret ServerMain.cxx**	Set **server** software for RPC authentication **(rpc_c_authn_dce_secret)**
8. Enter: **clearmake -C gnu**	Compile software from clearcase
9. Enter: **cp SleeperClient.cxx.authn_none SleeperClient.cxx**	Set **client** software for RPC authentication **(rpc_c_authn_none)**
10. Enter: **clearmake -C gnu**	Compile software from clearcase

11. Copy **ServerMain.cxx** and **SleeperClient.cxx** to your home directory.	
12. Login to <platform> workstation in **relacell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
13. cd **/usr/testa/IT/tools/CSS/SEC**	Location for test driver execution.
14. **ftp <workstation name> .hitc.com** ex. ftp casper.hitc.com	ftp to edfcell.
15. > **cd**	change directory to your home directory
16. > **ls**	List files in directory
17. > **bin**	binary mode
18. > **prompt**	disable prompting
19. > **mget S***	Transfers **Sleeper** files to the directory
20. Wait until all files are transferred	
21. > **quit**	exit ftp
22. Perform DCE login. **"dce_login <username>"**	
23. Enter DCE **password**	
24. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory.
25. Start application server. Enter: **server_authn_secret &**	
26. Verify that server is listening.	
27. Login to remote client.	Remote client login.
28. Perform DCE login. **"dce_login <username>"**	
29. Enter DCE **password**	
30. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory on client.
31. Start application client. Enter **client_authn_none**.	Client will login into server.
32. Verify client successfully rejects authentication attempt.	ERROR: DCE DCEException: authentication authorization mismatch (dce/rpc) ***C-CSS-21110***
33. Enter: **Kill -9 xxxx**	xxxx -> PID for **sleeper_server**
34. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	

4.1.2.23 Test Case 23: Mismatched Authorization (TC002.023)

The purpose of this test is to demonstrate that a client set for No-authorization will have its RPC request denied when the server is set for Authorization checking.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: User login name and password

Tools: OODCE 1.0.3, RogueWave Tools.h++ 6.1

Test Input:

Inputs to this test case include any valid DCE userid along with a server defined for Authorization checking and a client defined for No-authorization.

Test Output:

The expected results of this test case is a successful RPC process request rejection, by server.

Success Criteria

This test will be deemed successful when the server successfully denies, client RPC process request.

Test Procedures:

Test Case ID: TC002.023	
Test Name: Mismatch Authorization (Authorization Protocol)	
Test Steps:	Comments
1. Login to <platform> workstation in **edfcell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
2. cleartool setview <username>	Setup user clearcase view
3. cd **/ecs/formal/CSS/REL_A**	
4. Enter: **source .buildrc**	
5. cd **/ecs/formal/CSS/REL_A/DOF/src/SEC/test**	
6. Enter: **clearmake -C gnu clean**	Cleanup old **header** and **object** files
7. Enter: **cp ServerMain.cxx.authz_dce ServerMain.cxx**	Set **server** software for RPC authorization **(rpc_c_authz_dce)**
8. Enter: **clearmake -C gnu**	Compile software from clearcase
9. Enter: **cp SleeperClient.cxx.authz_none SleeperClient.cxx**	Set **client** software for RPC authorization **(rpc_c_authz_none)**
10. Enter: **clearmake -C gnu**	Compile software from clearcase
11. Copy **ServerMain.cxx** and **SleeperClient.cxx** to your home directory.	
12. Login to <platform> workstation in **relacell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
13. cd **/usr/testa/IT/tools/CSS/SEC**	Location for test driver execution.
14. **ftp <workstation name> .hitc.com** ex. ftp casper.hitc.com	ftp to edfcell.
15. > **cd**	change directory to your home directory
16.> **ls**	List files in directory
17.> **bin**	binary mode
18. > **prompt**	disable prompting
19. > **mget S***	Transfers **Sleeper** files to the directory
20. Wait until all files are transferred	
21. > **quit**	exit ftp
22. Start application server. Enter: server &	Server will autolog itself and start listening.
23. Verify that server is started.	
24. Start application client. Enter **client**.	Client request authentication into server.

25. Verify client successfully rejects authentication attempt.	ERROR: DCE DCEException: authentication authorization mismatch (dce/rpc) ***C-CSS-21110***
26. Enter: **Kill -9 xxxx**	xxxx -> PID for **sleeper_server**
27. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	

4.1.2.24 Test Case 24: Authorization None (TC002.024)

The purpose of this test is to demonstrate that a server defined for No-authorization of RPCs will allow any authenticated user to process RPCs regardless of authorizations defined for the client.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: User login name and password

Tools: OODCE 1.0.3, RogueWave Tools.h++ 6.1

Test Input:

Inputs to this test case include a server defined for No-authorization and a client defined for DCE Privilege Attribute Certificate (PAC).

Test Output:

The expected results of this test case are a successful processing of client RPC request with server define for No-Authorization.

Success Criteria

This test will be deemed successful when the client RPC request are successfully processed by the server.

Test Procedures:

Test Case ID: TC002.024	
Test Name: Authorization None (Authorization Protocol)	
Test Steps:	Comments
1. Login to <platform> workstation in **edfcell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
2. cleartool setview <username>	Setup user clearcase view
3. cd **/ecs/formal/CSS/REL_A**	
4. Enter: **source .buildrc**	

5. cd **/ecs/formal/CSS/REL_A/DOF/src/SEC/test**	
6. Enter: **clearmake -C gnu clean**	Cleanup old **header** and **object** files
7. Enter: **cp ServerMain.cxx.authz_none ServerMain.cxx**	Set **server** software for RPC authorization **(rpc_c_authz_none)**
8. Enter: **clearmake -C gnu**	Compile software from clearcase
9. Enter: **cp SleeperClient.cxx.authz_dce SleeperClient.cxx**	Set **client** software for RPC authorization **(rpc_c_authz_dce)**
10. Enter: **clearmake -C gnu**	Compile software from clearcase
11. Copy **ServerMain.cxx** and **SleeperClient.cxx** to your home directory.	
12. Login to <platform> workstation in **relacell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
13. cd **/usr/testa/IT/tools/CSS/SEC**	Location for test driver execution.
14. **ftp <workstation name> .hitc.com** ex. ftp casper.hitc.com	ftp to edfcell.
15. > **cd**	change directory to your home directory
16. > **ls**	List files in directory
17. > **bin**	binary mode
18. > **prompt**	disable prompting
19. > **mget S***	Transfers **Sleeper** files to the directory
20. Wait until all files are transferred	
21. > **quit**	exit ftp
22. Start application server. Enter: server &	Server will autolog itself and start listening.
23. Verify that server is started.	
24. Start application client. Enter **client**.	Client request authentication into server.
25. Verify RPC executed with no authorization checking.	***C-CSS-21110***
26. Enter: **Kill -9 xxxx**	xxxx -> PID for **sleeper_server**
27. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	

4.2 Basic Host/Application Management Tests

Basic Host and application management services are provided via an enterprise management system. This system is based on a manager-agent model consisting of management applications, a managed object model, and a management protocol. The management applications reside on managing systems. They provide the interfaces to perform management tasks. The managed object model consists of managed objects that represent the resources being managed, such as physical devices, system software, or applications. The MSS management agent is the implementation which substantiates the managed objects.

The basic host and application management service provides the following main functions:

- Enables the retrieval of ECS managed object values.
- Provides an extensible ECS management agent for managing host systems.
- Provides partial functionality for an extensible ECS management agent for managing ECS applications.

- Performs local polling on hosts to monitor the state of managed ECS resources.
- Performs event logging to the MSS local log file.
- Provides a uniform user interface and mechanism to monitor and control ECS managed object's attributes that are defined in the Management Information Base (MIB).
- Provides monitoring capabilities and event logging to the MSS local log file.

The following threads verify the functionality of the basic host and application management services:

- MIB/Extensible Agent-Host Thread (TC003)
- MIB/Extensible Agent - Application 1 Thread (TC004)
- Log File Management Thread (TC005)
- Monitor/Control Thread (TC006)

4.2.1 MIB/Extensible Host Thread (TC003)

The purpose of this thread is to verify:

- that MSS Management Agent can retrieve MIB data (i.e. object MIB attributes) from ECS managed objects. (ECS resources are defined in terms of a managed object.)
- that an extensible ECS management agent is provided for ECS applications.

Note: SNMP master agent is Peer Network's agent (a COTS product).

4.2.1.1 Test Case 1: MIB II Variable Validation (TC003.001)

The purpose of this test case is to query MIB II, which is the Host MIB, and Network device MIB information from the Host Resource Agent on Solaris and HP-UX machines by using the HP OpenView MIB browser. The MSS Monitor/Control Service is not complete as of phase 1; the MSS Management Agent Service only responds to requests for a managed object's MIB II attributes, and provides an agent for network devices. This test will be run on SUN and HP-UX platforms for phase 1.

<u>Test Configuration:</u>

Hardware: MSS Server, Managed resources: hosts (i.e. SUN, HP) and application(s), Inter networking subsystem components (i.e. LAN Network resources)

Software: HpOpenView, MSS-MACI components, Peer Network SNMP master agent on the host (MsAgAgent-COTS), MsAgEncps-COTS, MIB II Agent

Data: MIB definition

Tools: HpOpenView

Test Input:

Inputs to this test case include HpOpenView commands to examine the contents of the host MIB II.

Test Output:

Outputs include the MIB II contents displayed on the HpOpenView screen.

Success Criteria:

This test is successful if the MIB II values displayed via HpOpenView match those specified in the MIB definition documentation.

Test Procedures:

Test Case ID: TC003.001	
Test Name: MIB II Variable Validation	
Test Steps:	Comments:
1. Verify the snmp process is running by executing the following command on the SUN or HP-UX session: ps -ef | grep snmpd	Configuring and starting the Host Resource Agent on HP-UX (msse4hp) or SUN (msse3sun) platform**.**
2. If the snmpd process is listed, kill it by executing the following command: sudo kill -9 processid #	Note: This requires root or sudo root access.
3. Check the PEER master agent configuration file to check the manager(s) that may access the master agent.	The current included file "MsAgAgent.cfg" which is in /usr/testa/data directory will allow one machine, msse4hp (IP address 155.157.123.18)**,** to query the master agent.
4. Enter the PEER master agent directory: cd /usr/testa/bin	Starting the PEER master agent.
5. Start the master agent in the background: sudo ./MsAgAgent ../data/MsAgAgent.cfg filename ctrl z bg	Note: This requires root or sudo root access. Any filename is acceptable.
6. Start the encapsulator in the background: ./MsAgEncps -c ../data/MsAgEncps.cfg -s 1999 &	Starting the PEER Encapsulator.
7. To test **HP-UX** host resource agent, check and make sure the snmp port number is 1161 in the /etc/services file.	Note: If the HP-UX host resource agent connects to port 161, you must modify the snmp port to 1161 in the /etc/services file or if the machine is using NIS, you must modify the global NIS services file.
8. Start the **HP-UX** host resource agent: sudo /etc/snmpd & (This requires root or sudo root access.)	To start the **Solaris** host resource agent, type: sudo /opt/SUNWconn/snm/agents/snmpd -c /etc/opt/SUNWconn/snm/snmpd.conf -p 1161 & (Verify that the MSS Management Agent service provides agent for network devices - MSS-36070)

9. Query the Host Resource Agent Information by starting the HP OpenView MIB browser (on msse4hp machine): /usr/OV/bin/xnmbrowser	It is assumed that the agent is running on the machine that you wish to query, and that HPOpenView has been properly configured to query the Host Resource Agents.
10. Enter the Host name (e.g., msse3sun or msse4hp) in the box entitled "Name or IP Address".	Also, enter the IP Address instead of the Host name (e.g., 155.157.123.18 for msse4hp or 155.157.123.20 for msse3sun) in the box entitled "Name or IP Address".
11. Enter "public" in the box entitled "Community Name".	
12. Go to the MIB branch entitled: .iso.org.dod.internet.mgmt.mib-2	
13. All sub-branches should now be available to query.	
14. Query all groups in MIB II (e.g. System, Interfaces, Address Translation (AT), IP, ICMP, TCP, UDP, snmp groups)	When the system group is queried, check and make sure that the correct Host Resource agent is selected. Verify ability to respond to requests for managed object MIB attributes - MSS-36020, C-HRD-32010
15. Modify the snmpd.conf file (change any of the syscontact, syslocation...)	Forcing known information into the host resource. The snmpd.conf file for SUN is at: /etc/opt/SUNWconn/snm/snmpd.conf The snmpd.conf file for HP-UX is at: /etc/snmpd.conf
16. If the snmpd process is listed, kill it by excuting the following command: sudo kill -9 processid #	Note: This requires root or sudo root access.
17. Restart the snmpd process: /opt/SUNWconn/snm/agents/snmpd -c /etc/opt/SUNWconn/snm/snmpd.conf -p 1161 &	
18. On HPOpenView MIB browser, go to the MIB branch entitled '.iso.org.dod.internet.mgmt' and start query 'system'.	Check and make sure that the modified information in the snmpd.conf file is reflected on the MIB Values field of the HP OpenView MIB browser.
19. After querying MIB II of the Host Resource Agent, check the process id number of the following running jobs: ps -ef | grep MsAg ps -ef | grep snmpd	
20. Then, kill the master agent (MsAgAgent), the encapsulator (MsAgEncps), and the snmpd process by executing the following command: sudo kill -9 processid #	Note: This requires root or sudo root access.

4.2.2 MIB/Extensible Agent - Application 1 Thread (TC004)

This test thread verifies the partial functionality of the management subagent. The subagent is primarily responsible for supporting the ECS application MIB. The purpose of this thread is to verify:

- that based on SNMP Get requests from the master agent, the management subagent can retrieve the appropriate MIB values.
- the subagent can instantiate the object responsible for local polling of resources on the host.

Note: SNMP agent is Peer Network's agent (a COTS product).

4.2.2.1 Test Case 1: Store Information to MSS classes (TC004.001)

This test case verifies the ability to store and retrieve information to/from the EcAgMetric, EcAgConfigMetric, EcAgFaultMetric, EcAgPerfMetric, EcAgTuple, EcAgEvent, MsAgPerfEvent, EcAgHostInfo, and MsAgTeProc. For phase 1, the MSS Management Agent service only provides an agent for ECS applications, tables for the classes listed above, and partially handles set/get functions. This test will be run on SUN and HP-UX platforms for phase 1.

Test Configuration:

Hardware: MSS Server, Managed resources: hosts (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MSS-MACI components: EcAgMetric, EcAgConfigMetric, EcAgFaultMetric,EcAgPerfMetric, EcAgTuple, EcAgEvent, MsAgPerfEvent, EcAgHostInfo, MsAgTeProc, EcAgException

Data: N/A

Tools: EcAgTestP1_1PS (test program)

Test Input:

Test input includes keyins to store information to MSS classes.

Test Output:

Outputs include stored information to the MSS classes.

Success Criteria:

This test is successful when correct information is stored in the MSS classes.

Test Procedures:

Test Case ID: TC004.001	
Test Name: Store Information to MSS classes	
Test Steps:	Comments:
1. To test the ability to store information to the MSS classes, type: ./EcAgTestP1_1PS	
2. From the main menu, enter 0 for EcAgEvent.	Note: If invalid data type is entered, an empty object is returned.

3. Enter category(0-6), type (any positive value), severity (0-6), subsys (0-6), msgnum, msg, subtype, csci, TransactionID, TransactionParentID, tuple type, tuple value.	Verify that the input information is stored/retrieved in/from this class, the MSS Management Agent provides agent for ECS applications (MSS-36090), provides tables and partially handles set/get functions (MSS-36050, MSS-36040) Note: - Mode, appid, progid, procid, instanceid are hard code in phase 1.
4. Return to main menu, enter 1 for MsAgPerfEvent.	
5. Enter category, type, severity, subsys, msgnum, msg, subtype, csci, TransactionID, TransactionParentID, perf type, perf value, perf fault threshold, perf max threshold, perf min threshold, perf rearm fault threshold, perf rearm max threshold, perf rearm min threshold.	Verify that the input information is stored/retrieved in/from this class by comparing the information shown on the screen with the user input. They must matched. The MSS Management Agent provides agent for ECS applications (MSS-36090), provides tables and partially handles set/get functions (MSS-36050, MSS-36040) Note: - Mode, appid, progid, procid, instanceid are hard code in phase 1.
6. Return to main menu, enter 2 for EcAgPerfMetric.	
7. Enter perf type, perf value, perf fault threshold, perf max threshold, perf min threshold, perf rearm fault threshold, perf rearm max threshold, perf rearm min threshold	Verify that the input information is stored/retrieved in/from this class by comparing the information shown on the screen with the user input. They must matched. The MSS Management Agent provides agent for ECS applications (MSS-36090), provides tables and partially handles set/get functions (MSS-36050, MSS-36040)
8. Return to main menu, enter 3 for EcAgConfigMetric.	
9. Enter config type and config value.	Verify that the input information is stored/retrieved in/from this class by comparing the information shown on the screen with the user input. They must matched. The MSS Management Agent provides agent for ECS applications (MSS-36090), provides tables and partially handles set/get functions (MSS-36050, MSS-36040)
10. Return to main menu, enter 4 for EcAgFaultMetric.	
11. Enter fault type.	Verify that the input information is stored/retrieved in/from this class by comparing the information shown on the screen with the user input. They must matched. The MSS Management Agent provides agent for ECS applications (MSS-36090), provides tables and partially handles set/get functions (MSS-36050, MSS-36040)
12. Return to main menu, enter 5 for EcAgHostInfo.	Verify that the Host information is retrieved by system calls (correct hostname, ipaddr, osname...), the MSS Management Agent provides agent for ECS applications (MSS-36090), provides tables and partially handles set/get functions (MSS-36050, MSS-36040)

13. Return to main menu, enter 6 for selecting MsAgTeProc.	Verify that the Process information is retrieved by system calls (correct dsTTY.val, nUID ..., some process information is hard code in phase 1, e.g, nIndex, nInstanceID, nParentID, dsUpTime.val, nMode, dsTimeToStop.val...), the MSS Management Agent provides agent for ECS applications (MSS-36090), provides tables and partially handles set/get functions (MSS-36050, MSS-36040)

4.2.2.2 Test Case 2: Filter Events (TC004.002)

This test case verifies the ability to filter events based on severity level. For phase 1, the MSS Management Agent provides an agent for ECS applications and sends events received from the application to the subagent. This test will be run on SUN and HP-UX platforms for phase 1.

Test Configuration:

Hardware: MSS Server, Managed resources: hosts (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MSS-MACI components: EcAgManager, MsAgEventMgr, ecsApp.AppCfg,ecsApp.EcAgCfg, EcAgException

Data: N/A

Tools: cdsbrowser-COTS, ecsApp, subApp, ecsthrd (same as ecsApp, the difference is that ecsthrd sends 10 threads to send event to the subagent)

Test Input:

Inputs to this test case include entering the name of the application log file ('ecsApp.Applog') to the ecsApp.AppCfg configuration file and entering LogLevel, EventLevel, and UpdateInterval into the ecsApp.EcAgCfg configuration file. The user inputs the severity level to the menu entry.

Test Output:

Outputs include filtered events based on severity level.

Success Criteria:

The event is successfully sent to the subagent when the event's severity level is equal to or greater than the EventLevel in the ecsApp.EcAgCfg configuration file.

Test Procedures:

Test Case ID: TC004.002	
Test Name: Filter Events	
Test Steps:	Comments:
1. Check that two configuration files (ecsApp.AppCfg and ecsApp.EcAgCfg)exist on the same directory as the test programs (ecsApp and subApp).	ecsApp.AppCfg - This file contains the name of the application log file. The actual name of this file is 'ecsApp.Applog'. ecsApp.EcAgCfg - agent specific configurations. This file contains LogLevel (0 for disable logging events, 1 for enable logging events to /tmp/mss.log), EventLevel (0-6, filters based on severity level to determine whether to send the event to the subagent. The event's severity level must be equal to or greater than the EventLevel to be sent to the subagent.), and UpdateInterval (not used in phase 1 but value must be entered).
2. In the ecsApp.EcAgCfg file, set Eventlevel to 1.	Note: If you change the configuration items in the ecsApp.EcAgCfg file, you must first shut ecsApp down if it's up and running and restart it for ecsApp to use the new configurations.
3. Create one session (open a window in an x-term) and log into DCE by typing : dce_login	
4. Set environment variable: setenv CDS_SERVER_NAME /.:/subsys/HP/sample-apps/EcAgent	EcAgent is a default name, you can enter any name you want. Type 'echo' to make sure that the environment is created.
5. Start up subApp by typing: ./subApp and wait for "About to Listen for events" message.	
6. Create another session (open another window in an x-term) and log into DCE by typing: dce_login	Type 'echo' to make sure that the environment is created.
7. Set environment variable, CDS_SERVER_NAME to the same name as in step 4.	
8. Start up ecsApp by typing: ./ecsApp	
9. To create an event, enter the mandatory fields of an event (eventCategory: 0-6, eventType: any value is accepted for phase 1, severity: 0-6, subsys: 0-6).	
10. Enter 2 as severity level.	Since the event's severity level is higher than the EventLevel (1) set in the ecsApp.AppCfg file, the event is sent to the subagent. The subagent test program, subApp, then prints out the type, timestamp on the screen, and the rest of the event's attribute into file 'out'. Verify that MSS Management Agent Service provides agent for ECS applications and sends events received from application to subagent (MSS-36040, MSS-36090).

11. Enter 0 as severity level.	Since the event's severity level is lower than the EventLevel set in the ecsApp.AppCfg file, the event is not sent to the subagent. No print out is received for this case. Verify that MSS Management Agent Service provides agent for ECS applications and sends events received from application to subagent (MSS-36040, MSS-36090).
12. Enter 1 as severity level.	Since the event's severity level is equal to the EventLevel set in the ecsApp.AppCfg file, the event is also sent to the subagent. The subagent test program, subApp, also prints out the type, timestamp on the screen, and the rest of the event's attribute into file 'out'. Verify that MSS Management Agent Service provides agent for ECS applications and sends events received from application to subagent (MSS-36040, MSS-36090).
13. If you want to start the subApp again, first terminate the subApp by opening another window (msse4hp) and doing: ps -ef | grep username kill -9 process id # (of subApp) Next, make sure to clean up the CDS with this entry. You can do this by following the directions below: - Log into msse4hp - Log into DCE (dce_login) - setenv DISPLAY machineaddress:0.0 - Start up cdsbrowser by typing: cdsbrowser - Select "/.:/subsys", then "/.:subsys/HP" - Select "/.:/subsys/HP/sample-apps" - Select the object name that was registered. The default is: "/.:/subsys/HP/sample-apps/agent1" - Highlight the entry and press the right button of your mouse to get a menu. Select "delete entry" to remove the entry from CDS. - You may get out of cdsbrowser by select File, then Exit. Remember when logging out of DCE, type kdestroy.	Note: This is a bug in DCE.

4.2.2.3 Test Case 3: Enable & Disable Logging Events (TC004.003)

This test case verifies the ability to enable and disable logging events to mss.log. For phase 1, the MSS Management Agent sends events received from an application to the subagent. This test will be run on SUN and HP-UX platforms for phase 1.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MSS-MACI components: EcAgManager, MsAgEventMgr, ecsApp.AppCfg, ecsApp.EcAgCfg, EcAgException

Data: N/A

Tools: cdsbrowser-COTS, ecsApp, subApp, ecsthrd (same as ecsApp, the difference is that ecsthrd sends 10 threads to send event to the subagent)

Test Input:

Inputs to this test case include entering the name of the application log file ('ecsApp.Applog') to the ecsApp.AppCfg configuration file and entering LogLevel, EventLevel, and UpdateInterval into the ecsApp.EcAgCfg configuration file.

Test Output:

Outputs include logging events to mss.log.

Success Criteria:

This test is successful when events are either logged or unlogged to mss.log based on the Enable/Disable LogLevel set in the ecsApp.EcAgCfg configuration file.

Test Procedures:

Test Case ID: TC004.003	
Test Name: Enable & Disable Logging Events.	
Test Steps:	Comments:
1. Check that two configuration files (ecsApp.AppCfg and ecsApp.EcAgCfg) exist on the same directory as the test programs (ecsApp and subApp).	ecsApp.AppCfg - This file contains the name of the application log file. The actual name of this file is 'ecsApp.Applog'. ecsApp.EcAgCfg - agent specific configurations. This file contains LogLevel (0 for disable logging events, 1 for enable logging events to /tmp/mss.log) , EventLevel (0-6, filters based on severity level to determine whether to send the event to the subagent. The event's severity level must be equal to or greater than the EventLevel to be sent to the subagent.), and UpdateInterval (not used in phase 1 but value must be entered).
2. Check the ecsApp.EcAgCfg file and make sure that the LogLevel is set to 1 for enable logging events to /tmp/mss.log.	Note: If you change the configuration items in the ecsApp.EcAgCfg file, you must first shut ecsApp down if it's up and running and restart it for ecsApp to use the new configurations.
3. Create one session (open a window in an x-term) and log into DCE by typing : dce_login	
4. Set environment variable: setenv CDS_SERVER_NAME /.:/subsys/HP/sample-apps/EcAgent	EcAgent is a default name, you can enter any name you want. Type 'echo' to make sure that the environment is created.

5. Start up subApp by typing: ./subApp and wait for "About to Listen for events" message.	
6. Create another session (open another window in an x-term) and log into DCE by typing: dce_login	Type 'echo' to make sure that the environment is created.
7. Set environment variable, CDS_SERVER_NAME to the same name as in step 4.	
8. Start up ecsApp by typing: ./ecsApp	
9. To create an event, enter the mandatory fields of an event (eventCategory: 0-6, eventType: any value is accepted for phase 1, severity: 0-6, subsys: 0-6).	An event is created and logged to the /tmp/mss.log. Verify that the MSS Management Agent sends events received from application to subagent (MSS-36040).
10. In the ecsApp.EcAgCfg file, set LogLevel to 0 for disable logging events to the /tmp/mss.log.	Note: If you change the configuration items in the ecsApp.EcAgCfg file, you must first shut ecsApp down if it's up and running and restart it for ecsApp to use the new configurations.
11. Create one session (open a window in an x-term) and log into DCE by typing : dce_login	
12. Set environment variable: setenv CDS_SERVER_NAME /.:/subsys/HP/sample-apps/EcAgent	EcAgent is a default name, you can enter any name you want. Type 'echo' to make sure that the environment is created.
13. Start up subApp by typing: ./subApp and wait for "About to Listen for events" message.	
14. Create another session (open another window in an x-term) and log into DCE by typing: dce_login	Type 'echo' to make sure that the environment is created.
15. Set environment variable, CDS_SERVER_NAME to the same name as in step 4.	
16. Start up ecsApp by typing: ./ecsApp	
17. To create an event, enter the mandatory fields of an event (eventCategory: 0-6, eventType: any value is accepted for phase 1, severity: 0-6, subsys: 0-6).	The event is not logged to the /tmp/mss.log. Verify the MSS Management Agent sends events received from application to subagent (MSS-36040).

4.2.2.4 Test Case 4: Order Events (TC004.004)

This test case verifies the ability to allow events to always be sent and logged when a value of 6 (Order event) is entered to the eventCategory field. For phase 1, the MSS Management Agent sends events received from an application to the subagent. This test will be run on SUN and HP-UX platforms for phase 1.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MSS-MACI components: EcAgManager, MsAgEventMgr, ecsApp.AppCfg, ecsApp.EcAgCfg, EcAgException

Data: N/A

Tools: cdsbrowser-COTS, ecsApp, subApp, ecsthrd (same as ecsApp, the difference is that ecsthrd sends 10 threads to send events to the subagent)

Test Input:

Inputs to this test case include entering the name of the application log file ('ecsApp.Applog') to the ecsApp.AppCfg configuration file and entering LogLevel, EventLevel, and UpdateInterval into the ecsApp.EcAgCfg configuration file.

Test Output:

Outputs include order events sent to the subagent and logged to the mss.log.

Success Criteria:

The event is always sent to the subagent and logged even if the event's severity level is less than the EventLevel in the ecsApp.EcAgCfg configuration file.

Test Procedures:

Test Case ID: TC004.004	
Test Name: Order Events	
Test Steps:	Comments:
1. Check that two configuration files (ecsApp.AppCfg and ecsApp.EcAgCfg) exist on the same directory as the test programs (ecsApp and subApp).	ecsApp.AppCfg - This file contains the name of the application log file. The actual name of this file is 'ecsApp.Applog'. ecsApp.EcAgCfg - agent specific configurations. This file contains LogLevel (0 for disable logging events, 1 for enable logging events to /tmp/mss.log) , EventLevel (0-6, filters based on severity level to determine whether to send the event to the subagent. The event's severity level must be equal to or greater than the EventLevel to be sent to the subagent.), and UpdateInterval (not used in phase 1 but value must be entered).
2. Create one session (open a window in an x-term) and log into DCE by typing : dce_login	
3. Set environment variable: setenv CDS_SERVER_NAME /.:/subsys/HP/sample-apps/EcAgent	EcAgent is a default name, you can enter any name you want. Type 'echo' to make sure that the environment is created.

4. Start up subApp by typing: ./subApp and wait for "About to Listen for events" message.	
5. Create another session (open another window in an x-term) and log into DCE by typing: dce_login	Type 'echo' to make sure that the environment is created.
6. Set environment variable, CDS_SERVER_NAME to the same name as in step 4.	
7. Start up ecsApp by typing: ./ecsApp	
8. To create an event, enter the mandatory fields of an event (eventCategory: 0-6, eventType: any value is accepted for phase 1, severity: 0-6, subsys: 0-6).	
9. When the test program, ecsApp, prompts for EventCategory, enter 6.	6 means ordered event is selected.
10. When the test program prompts for severity level, enter different severity levels (0 to 6).	Events are always sent to the subagent and logged to the /tmp/mss.log even the severity level is less than the EventLevel set in the ecsApp.EcAgCfg file and the LogLevel is set to 0 for disable logging. Verify that the MSS Management Agent sends events received from application to subagent (MSS-36040).

4.2.2.5 Test Case 5: Create Events and Check for Valid Event's Attribute & Configuration fields (TC004.005)

This test case verifies the ability to create events and to check for valid event's attribute and configuration fields. For phase 1, the MSS Management Agent sends events received from application to subagent. This test will be run on SUN and HP-UX platforms for phase 1.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MSS-MACI components: EcAgManager, MsAgEventMgr, ecsApp.AppCfg, ecsApp.EcAgCfg, EcAgException

Data: N/A

Tools: cdsbrowser-COTS, ecsApp, subApp, ecsthrd (same as ecsApp, the difference is that ecsthrd sends 10 threads to send event to the subagent)

Test Input:

Inputs to this test case include entering the name of the application log file ('ecsApp.Applog') to the ecsApp.AppCfg configuration file, entering LogLevel, EventLevel, and UpdateInterval into the ecsApp.EcAgCfg configuration file, and entering eventCategory, eventType, severity, and subsys fields for events.

Test Output:

Outputs include created events, error messages for invalid event's attribute, and configuration fields.

Success Criteria:

Events are successfully created and error messages for invalid event's attribute and configuration fields are reported correctly.

Test Procedures:

Test Case ID: TC004.005	
Test Name: Create Events and Check for Valid Event's Attribute & Configuration fields	
Test Steps:	Comments:
1. Check that two configuration files (ecsApp.AppCfg and ecsApp.EcAgCfg) exist on the same directory as the test programs (ecsApp and subApp).	ecsApp.AppCfg - This file contains the name of the application log file. The actual name of this file is 'ecsApp.Applog'. ecsApp.EcAgCfg - agent specific configurations. This file contains LogLevel (0 for disable logging events, 1 for enable logging events to /tmp/mss.log) , EventLevel (0-6, filters based on severity level to determine whether to send the event to the subagent. The event's severity level must be equal to or greater than the EventLevel to be sent to the subagent.), and UpdateInterval (not used in phase 1 but value must be entered).
3. Create one session (open a window in an x-term) and log into DCE by typing : dce_login	
4. Set environment variable: setenv CDS_SERVER_NAME /.:/subsys/HP/sample-apps/EcAgent	EcAgent is a default name, you can enter any name you want. Type 'echo' to make sure that the environment is created.
5. Start up subApp by typing: ./subApp and wait for "About to Listen for events" message.	
6. Create another session (open another window in an x-term) and log into DCE by typing: dce_login	Type 'echo' to make sure that the environment is created.
7. Set environment variable, CDS_SERVER_NAME to the same name as in step 4.	
8. Start up ecsApp by typing: ./ecsApp	
9. To create an event, enter the mandatory fields of an event (eventCategory: 0-6, eventType: any value is accepted for phase 1, severity: 0-6, subsys: 0-6).	

10. For the eventCategory, severity level, and subsys fields, enter the values (0-6). For eventType, any value can be entered for phase 1.	Events with correct categories, severity levels, and subsys values are created. Verify that the MSS Management Agent sends events received from application to subagent (MSS-36040).
11. For eventCategory, severity level, and subsys fields, enter invalid values (greater than 6 or less than 0).	Error messages are shown on the screen.

4.2.2.6 Test Case 6: Functionality of EcAgConfigFile & MsAg Classes (TC004.006)

This test case verifies the functionality of the following classes:

- MsAgTblEntry
- MsAgTeAssoc
- MsAgTeChildInfo
- MsAgTeExec
- MsAgTeVerExec
- MsAgStaticBuffer
- MsAgSNMPTbl
- EcAgConfigFile (this class creates the EcAgConfigMetric, the EcAgFaultMetric, and the EcAgPerfMetric)
- MsAgTeApp

The MSS Monitor/Control Service is not complete as of phase 1; the MSS Management Agent Service only responds to requests for managed objects MIB II attributes. This test will be run on SUN and HP-UX platforms for phase 1.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MSS-MACI components: MsAgTblEntry, MsAgTeAssoc, MsAgTeChildInfo, MsAgTeExec, MsAgTeVerExec, MsAgStaticBuffer, MsAgSNMPTbl, EcAgConfigFile, MsAgTeApp, EcAgException

Data: test2.cfg

Tools: MsAgTestP1_1CK, MsAgTestP1_2CK (test programs)

Test Input:

Test input includes test2.cfg file.

Test Output:

Outputs include printout values of both objects 1 and 2 of the MsAgTblEntry, MsAgTeAssoc, MsAgTeChildInfo, MsAgTeExec, MsAgTeVerExec, MsAgStaticBuffer, MsAgSNMPTbl, MsAgTeApp classes, report of a cache miss, and printout values of the new configuration file.

Success Criteria:

This test is successful when the values of object 1 and 2 are identical.

Test Procedures:

Test Case ID: TC004.006	
Test Name: Functionality of the EcAgConfigFile & MsAg Classes	
Test Steps:	Comments:
1. To test the MsAg classes, type: ./MsAgTestP1_1CK	
a. MsAgTblEntry class	Two MsAgTblEntry objects are created. All values are set in object 2. The values are then copied through sets and gets into object 1. The values of each of both objects 1 and 2 are printed. They must be identical. To test the caching feature, the SetValues method is called, the test script sleeps for 3 secs, then calls the set values method again. The class should report a cache miss. (MSS-36020, 36050)
b. MsAgTeAssoc class	Two MsAgTeAssoc objects are created. All values are set in object 2. The values are then copied through sets and gets into object 1. The values of each of both objects 1 and 2 are printed. They must be identical. (MSS-36020, 36050)
c. MsAgTeChildInfo class	Two MsAgTeChildInfo objects are created. All values are set in object 2. The values are then copied through sets and gets into object 1. The values of each of both objects 1 and 2 are printed. They must be identical. (MSS-36020, 36050)
d. MsAgTeExec class	Two MsAgTeExec objects are created. All values are set in object 1. Object 1 values are then printed. The test script sleeps for 2 secs. The values are then copied through sets and gets into object 2. The values of object 2 are printed, and should be equal to that of object 1, with exception of uptime. (MSS-36020, 36050)
e. MsAgTeVerExec class	Two MsAgTeVerExec objects are created. All values are set in object 2. The values are then copied through sets and gets into object 1. The values of each of both objects 1 and 2 are printed. They must be identical. (MSS-36020, 36050)
f. MsAgStaticBuffer class	A MsAgStaticBuffer object is created. A stream of 7's are copied into the static buffer. The static buffer is then checked to see that the stream of 7's was entered correctly. Next, 100 strings are created and inserted into the static buffer. (MSS-36020, 36050)

g. MsAgSNMPTbl class	A MsAgSNMPTbl object is created. Ten threads are started that simultaneously insert MsAgTblEntries into the object. The threads then delete the items that they inserted. The table is printed to make sure that it is empty. (MSS-36020, 36050)
2. To test the EcAgConfigFile and MsAgTeApp classes, type: ./MsAgTestP1_2CK	
a. Test 1: EcAgConfigFile	EcAgConfigFile object is created and mapped to the config file "test2.cfg". Note: The file "test2.cfg" must exist and be a valid mgmt configuration file. The application configuration metrics of the file are printed. The application configuration metric "Name"'s value is changed to "This is the new name". The values of the configuration file are reprinted. The file name that the object is mapped to is changed to "output.dat" and the configuration file is written again. The test2.cfg file is located in the same directory as the MsAgTestP1_2CK test driver. (MSS-36020, 36050)
b. Test 2: MsAgTeApp	Two MsAgTeApp objects are created. All the values of object 1 are set. The values of 1 are copied to 2 through sets and gets. The values of both the objects are printed. Note: They must be identical. (MSS-36020, 36050)
c. Test 3: MsAgTeProg	Two MsAgTeProg objects are created. All the values of object 1 are set. The values of 1 are copied to 2 through sets and gets. The values of both the objects are printed. Note: They must be identical. (MSS-36020, 36050)

4.2.2.7 Test Case 7: Functionality of EcAgShutdown, EcAgException, MsAgMetVector (TC004.007)

This test verifies the functionality of the EcAgShutdown, the EcAgException, and the MsAgMetVector classes. The EcAgShutdown class is used by developers to overload a graceful shutdown procedure. The EcAgException class is used internally by the agent code to throw an exception when an error occurs. The MsAgMetVector class is used to format metric vectors to a streamable data before making rpc or oodce call to the subagent. The subagent then uses this class to stream back the data to another MsAgMetVector object.

For phase 1, the MSS Management Agent service only provides an agent for ECS applications, sends events received from an application to the subagent, and responds to requests for managed object MIB attributes. This test will be run on SUN and HP-UX platforms for phase 1.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MSS-MACI components: MsAgMetVector, EcAgShutdown, EcAgException

Data: N/A

Tools: EcAgTestP1_1EH (test program)

Test Input:

Test input includes information requested by MSS classes. Shutdown times are requested by EcAgShutdown class, and exception number is requested by EcAgException class.

Test Output:

Outputs include the Application, COTS, Program, and Process shutdown times, the error number and the string associated with the error number, 3 EcAgPerfMetric classes stored in a new MsAgMetVector (to test that the MsAgMetVector can read/write objects such as EcAgPerfMetric).

Success Criteria:

This test is successful when the Application, Program, Process, and COTS functions return the correct shutdown times, and the EcAgException class returns the expected error number and string associated with that number. Also, the stored objects in the new MsAgMetVector are the same as the ones stored previously (to simulate data transfer across the network).

Test Procedures:

Test Case ID: TC004.007	
Test name: Functionality of EcAgShutdown, EcAgException, MsAgMetVector	
Test Steps:	Comments:
1. To test the EcAgShutdown, EcAgException, MsAgMetVector classes, type: ./EcAgTestP1_1EH	
2. Run 1st test: Enter number of seconds to shutdown Application and COTS.	- For Application, if time is less than or equal 10, Application () will return 10. If it is greater than 10 , application will return that value. - For COTS, if time is less than or equal 5, COTS will return 5. If it is greater than 5, COTS will return that value. - Process and Program always return -1 as a default value for a normal condition. Verify that MSS Management Agent service provides agent for ECS applications and sends events received from application to subagent (MSS-36040, MSS-36090).

3. Run 2nd test: Enter exception number (0-17).	It prints the error number and the string associated with that number on the screen by using the type-cast operator functions for EcAgException. Make sure that negative numbers do not cause core dump to the EcAgException. Verify that MSS Management Agent service provides agent for ECS applications and sends events received from application to subagent (MSS-36040, MSS-36090).
4. To verify the results, compare it with the enumerated type, EcTAgErrCode definitions in: /ecs/formal/MSS/REL_A/MACI/include/EcAgException.h and in: /ecs/formal/MSS/REL_A/MACI/include/EcAgException.cxx	
5. Run 3rd test to test the MsAgMetVector.	It creates 3 EcAgPerfMetric classes and stores them into MsAgMetVector. The vector is flattened into a buffer and restored back into another MsAgMetVector. Objects in this new vector should be the same as the 3 EcAgPerfMetric classes. Verify that MSS Management Agent service provides agent for ECS applications and sends events received from application to subagent (MSS-36040, MSS-36090).
6. To exit this program, hit (1) when prompted to quit.	

4.2.3 Log File Management Thread (TC005)

This thread verifies:

- the interface between the CSS event logger service (EcUtLoggerRelA) and the MSS subagent (MsAgSubAgents).
- the ability of the CSS event logger service to accept and record application information, time events, and predetermined logging levels.
- the ability of the MSS Management Data service to provide the capability for an application to append records to log files and load the log files.

4.2.3.1 Test Case 1: Transfer Scheduled Log Files (TC005.001)

This test case verifies the ability to transfer the individual management data log files from each managed host, to the MSS Server, on a scheduled basis, and to append log file entries into the prototype log file. This test will be run on SUN and HP-UX platforms for phase 1.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MSS-MDA components: MsMdConfigurationEntry, MsMdConfigurationList, MsMdScheduleEntry, MsMdSchedule, MsMdAggregateLogEntry, MsMdHostAggregateLogList, MsMdAggregateLogFileList, MsMdManager (Log Transfer)

Data: N/A

Tools: MsMdPhase1

Test Input:

Inputs to this test case include entering configuration entries manually.

Test Output:

Outputs include a configuration list, a schedule list, and log files in the MSS Server.

Success Criteria:

This test is successful when log files are transferred and stored in the MSS Server on a scheduled basis and log file entries can be appended to the log file.

Test Procedures:

Test Case ID: TC005.001	
Test Name: Transfer Scheduled Log Files	
Test Steps:	Comments:
1. On the test machine, type: setenv LogFileDir /usr/testa/results/	
2. Execute the test program by typing : ./MsMdPhase1	
3. It prompts you to enter username and password.	
4. From the main menu, enter 1 for insert. From the submenu, enter number of configuration entries, enter host name, enter **interval** (in minutes when the log file is to be transferred to MSS server, enter size limit (size of threshold at which the log file is to be transferred, not utilized for phase 1, any value is accepted), enter number of time, enter time (**absolute time** at which the log file is to be transferred).	Absolute time (date and time to transfer the log file are shown on the screen).
5. From the main menu, enter 3 to print configuration list.	Configuration list is shown on the screen (Host name, Interval, Size Limit, Absolute Time List).
6. From the main menu, enter 4 to print schedule list.	Schedule list is shown on the screen (Host name, date and time to transfer log file) (phase 1 only schedule log file - MSS-18260, MSS-18270, MSS-18280)
7. From the main menu, enter 2 to transfer the log file.	The message 'connection complete, file transferred, ftp complete' is shown on the screen (MSS-18050).

8. Go to the directory that contains the transferred log file: cd /usr/testa/results/	Make sure that the log file has been transferred and stored in this directory.
9. From the main menu, enter 5 to print aggregate list.	Aggregate list is shown on the screen (Host name, start time, end time, compressed, directory that the log file is sent to, file id, file size, file time stamp).
10. From the main menu, enter 2 to transfer another host log file.	
11. Go to the directory that contains the transferred log file: cd /usr/testa/results/	Make sure that the new record has been appended to the log file (MSS-18330).

4.2.4 Basic Host/Application Management Build (BC002)

The purpose of this build is to verify the partial functionality of the basic host/application management services.

4.2.4.1 Test Case 1: Initial Host Management Capabilities (BC002.001)

This test case verifies the ability to query Host MIB and Network device MIB (MIB II) from HpOpenView MIB browser.

Test Configuration:

Hardware: MSS Server, Managed resources: host(i.e. SUN, HP) and application(s), Inter networking subsystem components (i.e. LAN Network resources)

Software: HpOpenView, MSS-MACI components, Peer Network SNMP master agent on the host, (MsAgAgent-COTS), MsAgEncps-COTS, MIB II Agent

Data: MIB definition

Tools: HpOpenView

Test Input:

Inputs to this test case include HpOpenView commands to examine the contents of the host MIB II.

Test Output:

Outputs include the MIB II contents displayed on HpOpenView screen.

Success Criteria:

This test is successful if the MIB II values displayed via HpOpenView match those specified in the MIB definition documentation.

Test Procedures:

Test Case ID: BC002.001	
Test Name: Initial Host Management Capabilitiees	
Test Steps:	Comments:
1. Verify the snmp process is running by executing the following command on the SUN or HP-UX session: ps -ef | grep snmpd	Configuring and starting the Host Resource Agent on HP-UX (msse4hp) or SUN (msse3sun) platform**.**
2. If the snmpd process is listed, kill it by executing the following command: sudo kill -9 processid #	Note: This requires root or sudo root access.
3. Check the PEER master agent configuration file to reflect the manager(s) that may access the master agent.	The current included file "MsAgAgent.cfg" which is in /usr/testa/data directory will allow one machine, msse4hp (IP address 155.157.123.18)**,** to query the master agent.
4. Enter the PEER master agent directory: cd /usr/testa/bin	Starting the PEER master agent.
5. Start the master agent in the background: sudo ./MsAgAgent ../data/MsAgAgent.cfg filename ctrl z bg	Note: This requires root or sudo root access. Any filename is acceptable.
6. Start the encapsulator in the background: ./MsAgEncps -c ../data/MsAgEncps.cfg -s 1999 &	Starting the PEER Encapsulator.
7. To test **HP-UX** host resource agent, check and make sure the snmp port number is 1161 in the /etc/services file.	Note: If the HP-UX host resource agent connects to port 161, you must modify the snmp port to 1161 in the /etc/services file or if the machine is using NIS, you must modify the global NIS services file.
8. Start the **HP-UX** host resource agent: sudo /etc/snmpd & (This requires root or sudo root access.)	To start the **Solaris** host resource agent, type: sudo /opt/SUNWconn/snm/agents/snmpd -c /etc/opt/SUNWconn/snm/snmpd.conf -p 1161 & (Verify that the MSS Management Agent service provides agent for network devices - MSS-36070)
9. Query the Host Resource Agent Information by starting the HP OpenView MIB browser (on msse4hp machine): /usr/OV/bin/xnmbrowser	It is assumed that the agent is running on the machine that you wish to query, and that HP OpenView has been properly configured to query the Host Resource Agents.
10. Enter the Host name (e.g., msse3sun or msse4hp) in the box entitled "Name or IP Address".	Also, enter the IP Address instead of the Host name (e.g., 155.157.123.18 for msse4hp or 155.157.123.20 for msse3sun) in the box entitled "Name or IP Address".
11. Enter "public" in the box entitled "Community Name".	
12. Go to the MIB branch entitled: .iso.org.dod.internet.mgmt.mib-2	
13. All sub-branches should now be available to query.	
14. Query all groups in MIB II (e.g. System, Interfaces, Address Translation (AT), IP, ICMP, TCP, UDP, snmp groups)	When the system group is queried, check and make sure that the correct Host Resource agent is selected. Verify ability to respond to requests for managed object MIB attributes - MSS-36020, ISS-2110.

15. Modify the snmpd.conf file (change any of the syscontact, syslocation...)	Forcing known information into the host resource. The snmpd.conf file for SUN is at: /etc/opt/SUNWconn/snm/snmpd.conf The snmpd.conf file for HP-UX is at: /etc/snmpd.conf
16. If the snmpd process is listed, kill it by excuting the following command: sudo kill -9 processid #	Note: This requires root or sudo root access.
17. Restart the snmpd process: /opt/SUNWconn/snm/agents/snmpd -c /etc/opt/SUNWconn/snm/snmpd.conf -p 1161 &	
18. On HP OpenView MIB browser, go to the MIB branch entitled '.iso.org.dod.internet.mgmt' and start query 'system'.	Check and make sure that the modified information in the snmpd.conf file is reflected on the MIB Values field of the HP OpenView MIB browser.
19. After querying MIB II of the Host Resource Agent, check the process id number of the following running jobs: ps -ef | grep MsAg ps -ef | grep snmpd	
20. Then, kill the master agent (MsAgAgent), the encapsulator (MsAgEncps), and the snmpd process by executing the following command: sudo kill -9 processid #	Note: This requires root or sudo root access.

4.2.4.2 Test Case 2: Events Capabilities (BC002.002)

This test case verifies the ability to filter, enable/disable, order, and create events.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MSS-MACI components: EcAgManager, MsAgEventMgr, ecsApp.AppCfg, ecsApp.EcAgCfg

Data: N/A

Tools: ecsApp, subApp, ecsthrd (same as ecsApp, the difference is that ecsthrd sends 10 threads to send event to the subagent)

Test Input:

Inputs to this test case include entering the name of the application log file ('generic.log') to the ecsApp.AppCfg configuration file and entering LogLevel, EventLevel, UpdateInterval into the ecsApp.EcAgCfg configuration file, and entering eventCategory, eventType, eventSubType, severity fields for events.

Test Output:

Outputs include created events, filtered events based on severity level, logging events to mss.log, order events sent to the subagent, error messages for invalid event's attribute and configuration fields.

Success Criteria:

Events are successfully created, filtered based on severity level, logged/unlogged based on enable/disable LogLevel. Order event is always sent to the subagent and logged, and error messages for invalid event's attribute and configuration fields are reported correctly.

Test Procedures:

Test Case ID: BC002.002	
Test Name: Events Capabilities	
Test Steps:	Comments:
1. Check that two configuration files (ecsApp.AppCfg and ecsApp.EcAgCfg) must exist on the same directory as the test programs (ecsApp and subApp).	ecsApp.AppCfg - This file contains the name of the application log file. The actual name of this file is 'ecsApp.Applog'. ecsApp.EcAgCfg - agent specific configurations. This file contains LogLevel (0 for disable logging events, 1 for enable logging events to /tmp/mss.log) , EventLevel (0-6, filters based on severity level to determine whether to send the event to the subagent. The event's severity level must be equal to or greater than the EventLevel to be sent to the subagent.), and UpdateInterval (not used in phase 1 but value must be entered).
2. In the ecsApp.EcAgCfg file, set Eventlevel to 1.	Note: If you change the configuration items in the ecsApp.EcAgCfg file, you must first shut ecsApp down if it's up and running and restart it for ecsApp to use the new configurations.
3. Create one session (open a window in an x-term) and log into DCE by typing : dce_login	
4. Set environment variable: setenv CDS_SERVER_NAME /.:/subsys/HP/sample-apps/EcAgent	EcAgent is a default name, you can enter any name you want. Type 'echo' to make sure that the environment is created.
5. Start up subApp by typing: ./subApp and wait for "About to Listen for events" message.	
6. Create another session (open another window in an x-term) and log into DCE by typing: dce_login	Type 'echo' to make sure that the environment is created.
7. Set environment variable, CDS_SERVER_NAME to the same name as in step 4.	
8. Start up ecsApp by typing: ./ecsApp	
9. To create an event, enter the mandatory fields of an event (eventCategory: 0-6, eventType: any value is accepted for phase 1, severity: 0-6, subsys: 0-6).	

10. Enter 2 as severity level.	Since the event's severity level is higher than the EventLevel (1) set in the ecsApp.AppCfg file, the event is sent to the subagent. The subagent test program, subApp, then prints out the type, timestamp on the screen, and the rest of the event's attribute into file 'out'. Verify that MSS Management Agent Service provides agent for ECS applications and sends events received from application to subagent.
11. Enter 0 as severity level.	Since the event's severity level is lower than the EventLevel set in the ecsApp.AppCfg file, the event is not sent to the subagent. No print out is received for this case. Verify that MSS Management Agent Service provides agent for ECS applications and sends events received from application to subagent.
12. Enter 1 as severity level.	Since the event's severity level is equal to the EventLevel set in the ecsApp.AppCfg file, the event is also sent to the subagent. The subagent test program, subApp, also prints out the type, timestamp on the screen, and the rest of the event's attribute into file 'out'. Verify that MSS Management Agent Service provides agent for ECS applications and sends events received from application to subagent.
13. In the ecsApp.EcAgCfg file, set LogLevel to 0 for disable logging events to the /tmp/mss.log.	Note: If you change the configuration items in the ecsApp.EcAgCfg file, you must first shut ecsApp down if it's up and running and restart it for ecsApp to use the new configurations.
14. Restart ecsApp to use the new configurations by typing: ./ecsApp	
15. To create an event, enter the mandatory fields of an event (eventCategory: 0-6, eventType: any value is accepted for phase 1, severity: 0-6, subsys: 0-6).	The event is not logged to the /tmp/mss.log. Verify the MSS Management Agent sends events received from application to subagent.
16. When the test program, ecsApp, prompts for EventCategory, enter 6.	6 means ordered event is selected.
17. When the test program prompts for severity level, enter different severity levels (0 to 6).	Events are always sent to the subagent and logged to the /tmp/mss.log even the severity level is less than the EventLevel set in the ecsApp.EcAgCfg file and the LogLevel is set to 0 for disable logging. Verify that the MSS Management Agent sends events received from application to subagent.
18. For eventCategory, severity level, and subsys fields, enter invalid values (greater than 6 or less than 0).	Error messages are shown on the screen.

19. If you want to start the subApp again, first terminate the subApp by opening another window (msse4hp) and doing: ps -ef | grep username kill -9 process id # (of subApp) Next, make sure to clean up the CDS with this entry. You can do this by following the directions below: - Log into msse4hp - Log into DCE (dce_login) - setenv DISPLAY machineaddress:0.0 - Start up cdsbrowser by typing: cdsbrowser - Select "/.:/subsys", then "/.:subsys/HP" - Select "/.:/subsys/HP/sample-apps" - Select the object name that was registered. The default is: "/.:/subsys/HP/sample-apps/agent1" - Highlight the entry and press the right button of your mouse to get a menu. Select "delete entry" to remove the entry from CDS. - You may get out of cdsbrowser by select File, then Exit. Remember when logging out of DCE, type kdestroy.	Note: This is a bug in DCE.

4.2.4.3 Test Case 3: Transfer Scheduled Log Files Capabilities (BC002.003)

This test case verifies the ability to transfer the individual management data log files from each managed host, to the MSS Server, on a scheduled basis and to append log file entries into the prototype log file.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MSS-MDA components

Data: N/A

Tools: MsMdPhase1

Test Input:

Inputs to this test case include entering configuration entries manually.

Test Output:

Outputs include configuration list, schedule list, log files in the MSS Server.

Success Criteria:

This test is successful when log files are transferred and stored in the MSS Server on a scheduled basis and log file entries can be appended to the log file.

Test Procedures:

Test Case ID: BC002.003	
Test Name: Transfer Scheduled Log Files Capabilities	
Test Steps:	Comments:
1. On the test machine, type: setenv LogFileDir /usr/testa/results/	
2. Execute the test program by typing : ./MsMdPhase1	
3. It prompts you to enter username and password.	
4. From the main menu, enter 1 for insert. From the submenu, enter number of configuration entries, enter host name, enter **interval** (in minutes when the log file is to be transferred to MSS server, enter size limit (size of threshold at which the log file is to be transferred, not utilized for phase 1, any value is accepted), enter number of time, enter time (**absolute time** at which the log file is to be transferred).	Absolute time (date and time to transfer the log file are shown on the screen).
5. From the main menu, enter 3 to print configuration list.	Configuration list is shown on the screen (Host name, Interval, Size Limit, Absolute Time List).
6. From the main menu, enter 4 to print schedule list.	Schedule list is shown on the screen (Host name, date and time to transfer log file).
7. From the main menu, enter 2 to transfer the log file.	The message 'connection complete, file transferred, ftp complete' is shown on the screen.
8. Go to the directory that contains the transferred log file: cd /usr/testa/results/	Make sure that the log file has been transferred and stored in this directory.
9. From the main menu, enter 5 to print aggregate list.	Aggregate list is shown on the screen (Host name, start time, end time, compressed, directory that the log file is sent to, file id, file size, file time stamp).
10. From the main menu, enter 2 to transfer another host log file.	
11. Go to the directory that contains the transferred log file: cd /usr/testa/results/	Make sure that the new record has been appended to the log file.

4.3 Configuration Management Tests

The following threads verify the set of requirements related to configuration management:

- Trouble Ticketing Thread (TC007)
- Software CM Thread (TC008)
- Physical CM Thread (TC009)

4.3.1 Trouble Ticketing Thread (TC007)

The Trouble Ticketing (TT) service provides a consistent means of reporting, classifying, and tracking problem occurrence and resolution. Trouble Ticketing enables both end users and internal users to submit, forward, and query historical and current trouble tickets. (An end user

applies to users in remote locations, such as Distributed Active Archive Centers, who can submit trouble tickets but do not have access to The Remedy Action Request System. An internal user applies to administrative users, such as help-desk personnel, who can submit and access The Remedy Action System.)

The ECS Trouble Ticketing Service allows end users to submit and list trouble tickets. The Remedy Action Request System allows internal users to search, query, forward, and make modifications to trouble ticket status, pertinent information, and resolutions. The following test cases illustrate the interface between the ECS Trouble Ticket Service and The Remedy Action Request System.

4.3.1.1 Test Case 1: Notification of Status Changes (TC007.001)

This test case verifies that the originator of a trouble ticket is notified when the status of the submitted trouble ticket has been modified. The notification is via E-Mail.

Test Configuration:

Hardware: HP server

Software: Management Services (Remedy Package), E-Mail functionality, MsTtManager

Data: Sample problem to report, including a short problem description and severity number

Tools: Xrunner/Loadrunner

Test Input:

Inputs include the creation of a trouble ticket that is modified throughout the course of the test, and a simulated SMC E-Mail address.

Test Output:

Notification via E-Mail that the Trouble Ticket's status has been modified.

Success Criteria:

The test is successful when the Trouble Ticketing Service has automatically notified the user via E-mail regarding the change in status.

Test Procedures:

Test Case ID: TC007.001	
Test Name: Notification Of Status Changes	
Test Steps:	Comments:
1. Log onto ECS Trouble Ticket Service.	
2. Select submit a trouble ticket.	
3. Verify submit a trouble ticket homepage appears.	
4. Enter the User Information.	

5. Enter the Problem Information.	
6. Select submit icon.	
7. Return to ECS Trouble Ticket Service Home page.	
8. Log-off as user.	
9. Log-on as Internal user.	
10.Select Release-A Action Request System.	
11. Enter the ticket-id.	
12. Select query-list.	
13. Select query-modify individual.	
14.Verify that the Modify individual homepage appears.	
15. Modify the ticket status to closed.	C-MSS-57510
16. Select closing code.	
17. Select apply.	
18. Select Dismiss.	
19. Log-off as Internal user.	
20. Log-on as user.	
21.Verify that an E-Mail notification message has been received.	

4.3.1.2 Test Case 2: Historical and Current Trouble Tickets (TC007.002)

This test case verifies that the Trouble Ticketing Service enables an internal user to search for historical and current trouble tickets using keywords, user id, and trouble ticket id.

Test Configuration:

Hardware: HP server

Software: Management Services (Remedy Package), MsTtManager

Data: Sample problem to report, including a short problem description and severity number

Tools: Xrunner/Loadrunner

Test Input:

Inputs include the creation of a trouble ticket whose status is queried via keywords, User ID, and Ticket ID.

Test Output:

Historical and/or currents tickets retrieved as a result of a user's request.

Success Criteria:

The test is successful when the Trouble Ticketing Service has retrieved the requested trouble ticket utilizing keywords, user id or ticket id.

Test Procedures:

Test Case ID: TC007.002	
Test Name: Historical and Current Trouble Tickets	
Test Steps:	Comments:
1. Log onto ECS Trouble Ticket Service as Internal user.	
2. Perform a search for a historical trouble ticket utilizing keyword(s), user-id, trouble-ticket-id.	
3. Perform a search for a current trouble ticket utilizing keyword(s), user-id, trouble-ticket-id.	C-MSS-57530
4. Verify that the requested search for the current trouble ticket is successful.	
5. Verify that the requested search for the historical trouble ticket is successful.	

4.3.1.3 Test Case 3: Forwarding Trouble Tickets (TC007.003)

This test case verifies that the Trouble Ticketing Service allows the internal user to forward a selected trouble ticket from one organization to another.

Test Configuration:

Hardware: HP server

Software: Management Services (Remedy Package), MsTtManager

Data: Sample problem to report, including a short problem description and severity number

Tools: Xrunner/Loadrunner

Test Input:

Inputs include the creation of a trouble ticket that is forwarded to another organization, and a simulated SMC E-Mail address.

Test Output:

Notification of trouble ticket receipt.

Success Criteria:

The test is successful when the Trouble Ticketing Service has forwarded selected trouble tickets to another organization and receipt of delivery has been returned.

Test Procedures:

Test Case ID: TC007.003	
Test Name: Forwarding Trouble Tickets	
Test Steps:	Comments:
1. Log onto ECS Trouble Ticket Service as User.	
2. Submit a trouble ticket.	
3. Log onto ECS Trouble Ticket Service as Internal user.	
4. Enter ticket-id.	

5. Verify that the selected ticket-id was retrieved.	
6. Forward the trouble ticket to a technician, help-desk, or another site.	C-MSS-57540
7. Enter data into forward-to, forward-from, forward-by, and forward-date cells.	
8. Log-off as internal user.	
9. Log-on as technician.	
10. Verify that the selected trouble ticket was forwarded.	

4.3.1.4 Test Case 4: Trouble Tickets Status (TC007.004)

This test case verifies that the Trouble Ticketing Service maintains different trouble ticket status's to include Open, Work-In-Process, Closed, and Archived.

Test Configuration:

Hardware: HP server

Software: Management Services (Remedy Package), MsTtManager

Data: Sample problem to report, including a short problem description and severity number

Tools: Xrunner/Loadrunner

Test Input:

Inputs include the creation of several trouble tickets with varying status descriptions, including Open, Work-in-Process, Closed, and Archived.

Test Output:

Notification of the requested trouble ticket's status.

Success Criteria:

The test is successful when the Trouble Ticketing Service has notified the user of the correct status of the requested trouble ticket.

Test Procedures:

Test Case ID: TC007.004	
Test Name: Trouble Tickets Status	
Test Steps:	Comments:
1. Log onto ECS Trouble Ticket Service as user.	
2. Submit several trouble tickets.	
3. Log-off as user.	
4. Log-on as Administrator, or help-desk.	
5. Search and modify one of the trouble ticket status to work-in-process.	C-MSS-57550
6. Select apply.	
7. Search and modify one of the trouble ticket status to closed.	
8. Select apply.	
9. Search and modify one of the trouble ticket status to open.	

10. Select apply.	
11. Search and modify one of the trouble ticket status to archived.	
12. Select apply.	
13. Select dismiss.	
14. Return to the Action Request System.	
15. Query the selected trouble tickets.	
16. Verify that the ECS Trouble Ticket Service maintained the different ticket status's.	

4.3.1.5 Test Case 5: Trouble Ticket Search (TC007.005)

This test case verifies that the Trouble Ticketing Service provides the internal user the ability to search for trouble tickets relating to the same source.

Test Configuration:

Hardware: HP server

Software: Management Services (Remedy Package), MsTtManager

Data: Sample problem to report, including a short problem description and severity number

Tools: Xrunner/Loadrunner

Test Input:

Inputs include the creation of several trouble tickets pertaining to the same piece of equipment.

Test Output:

Notification that the search produced trouble tickets all related to the selected equipment.

Success Criteria:

The test is successful when the Trouble Ticketing Service has searched and retrieved all trouble tickets related to the selected equipment.

Test Procedures:

Test Case ID: TC007.005	
Test Name: Trouble Ticket Search	
Test Steps:	Comments:
1. Log-on to ECS Trouble Ticket Service as a user.	
2. Select submit a trouble ticket.	
3. Submit several trouble tickets concerning a selected piece of hardware utilizing the hardware resource field.	C-MSS-57560
4. Log-off as user.	
5. Log-on as internal user.	
6. Select hardware resource icon.	
7. Select the hardware in the list that is associated with your trouble tickets.	
8. Select query-list.	

9. Verify that search on the selected piece of hardware is displayed.	
10. Log-off of the Remedy.	

4.3.1.6 Test Case 6: Trouble Ticket Storage and Retrieve (TC007.006)

This test case verifies that the Trouble Ticketing Service is able to store information that is pertinent to trouble tickets, and retrieve this information at a later time.

Test Configuration

Hardware: HP server

Software: Management Services (Remedy Package), MsTtManager

Data: Sample problem to report, including a short problem description and severity number

Tools: Xrunner/Loadrunner

Test Input:

Inputs include the creation of a trouble ticket that includes keywords, description, problem solution, and originator.

Test Output:

The unique trouble ticket ID, status, description, associated resources, problem solution, originator, and keywords.

Success Criteria:

The test is successful when the Trouble Ticketing Service has searched and retrieved stored information for a selected trouble ticket. The Trouble Ticketing service must include the unique trouble ticket ID, status, description, associated resources, problem solution, originator, and keywords.

Test Procedures:

Test Case ID: TC007.006	
Test Name: Trouble Ticket Storage and Retrieve	
Test Steps:	Comments:
1. Log onto ECS Trouble Ticket Service as an internal user.	
2. Enter User Information.	C-MSS-57580
3. Enter Problem Information.	
4. Enter Short Description.	
5. Enter Detailed problem description.	
6. Return to ECS Trouble Ticket Service.	
7. Select list trouble ticket.	
8. Select a trouble ticket that was submitted.	

9. Verify that ECS Trouble Ticket Service was able to retrieve the data at a later time.	
10. Log-off as user.	

4.3.1.7 Test Case 7: Trouble Ticket Reports (TC007.007)

This test case verifies that the Trouble Ticketing Service allows the user to generate reports from its database and print to a screen or printer.

Test Configuration:

Hardware: HP server

Software: Management Services (Remedy Package), MsTtManager

Data: Sample problem to report, including a short problem description and severity number

Tools: Xrunner/Loadrunner

Test Input:

Inputs include the creation of a trouble ticket report that is displayed and printed.

Test Output:

Report of selected trouble ticket, both displayed and printed.

Success Criteria:

The test is successful when the Trouble Ticketing Service has allowed the user to generate reports from its database and print to the screen or printer.

Test Procedures:

Test Case ID: TC007.007	
Test Name: Trouble Ticket Reports	
Test Steps:	Comments:
1. Log onto ECS Trouble Ticket Service.	
2. Select Query-list to display a list of Trouble Tickets.	
3. Select a Trouble Ticket.	
4. Query report of Trouble Ticket selected.	
5. Verify report was generated.	C-MSS-57610
6. Select print to screen or printer.	C-MSS-57620
7. Verify report was printed to screen or printer.	
8. Log-off.	

4.3.1.8 Test Case 8: Trouble Ticket Escalation (TC007.008)

This test case verifies that the Trouble Ticketing Service provides customization features that enable internal users from other locations to specify notification and escalation rules.

Test Configuration:

Hardware: HP server

Software: Management Services (Remedy Package), MsTtManager

Data: Sample problem to report, including a short problem description and severity number

Tools: Xrunner/Loadrunner

Test Input:

Inputs include the creation of a trouble ticket whose priority is modified throughout the course of the test.

Test Output:

Notification of the selected trouble ticket's priority change.

Success Criteria:

The test is successful when the Trouble Ticketing Service has allowed the users at various locations to prioritize their notifications. The users must to be able to change the priority of a trouble ticket at any time and affected users must be notified.

Test Procedures:

Test Case ID: TC007.008	
Test Name: Trouble Ticket Escalation	
Test Steps:	Comments:
1. Log onto ECS Trouble Ticket Service as internal user.	
2. Query a selected trouble ticket.	
3. Enter Problem Information.	
4. Enter Short Description.	
5. Enter Detailed problem description.	
6. Log-off as user.	
7. Log-on as Internal user.	
8. Query the trouble ticket that was recently submitted.	C-MSS-57630
9. Modify the assigned-priority status.	

4.3.1.9 Test Case 9: Graphical User Interface (TC007.009)

This test case verifies that the Trouble Ticketing Service provides a graphical user interface to support the entry, submission, and editing of trouble tickets by a registered user.

Test Configuration:

Hardware: HP server

Software: Management Services (Remedy Package), MsTtManager, MsTtUserInterface

Data: Sample problem to report, including a short problem description and severity number

Tools: Xrunner/Loadrunner

Test Input:

Inputs include the creation, submission, and modification of a trouble ticket through a GUI interface.

Test Output:

Screen output displaying GUI interface and user's inputs.

Success Criteria:

The test is successful when a registered user is allowed to enter the Trouble Ticketing Service and modify the selected trouble ticket via the GUI.

Test Procedures:

Test Case ID: TC007.009	
Test Name: Graphical User Interface	
Test Steps:	Comments:
1. Log onto ECS Trouble Ticket Service.	
2. Select submit a trouble ticket.	
3. Verify submit a trouble ticket homepage appears.	
4. Enter the User Information.	C-MSS-57600
5. Enter the Problem Information.	
6. Enter Short Description.	
7. Enter Detailed Description.	
8. Enter Detailed problem description.	
9. Select submit icon.	
10. Return to ECS Trouble Ticket Service Home page.	
11. Select list trouble tickets.	
12. Query a selected trouble ticket.	
13. Modify information about the trouble ticket.	C-MSS-57500
14. Return to Trouble Ticket Homepage.	
15. Select list trouble tickets.	
16. Select trouble ticket previously modified.	
17. Verify that the trouble ticket changes were saved.	
18. Log-off of the ECS Trouble Ticket Service.	

4.3.1.10 Test Case 10: Application Program Interface (TC007.010)

This test case verifies that the Trouble Ticketing Service enables the user to enter trouble tickets from another application program.

Test Configuration:

Hardware: HP server

Software: Management Services (Remedy Package),

Data: Sample problem to report, including a short problem description and severity number

Tools: Xrunner/Loadrunner

Test Input:

Trouble Ticket and Application Package.

Test Output:

Notification that the trouble ticket was created and submitted by another application program.

Success Criteria:

The test is successful when the Trouble Ticketing Service has allowed the user to create and submit a trouble ticket from another application program.

Test Procedures:

Test Case ID: TC007.010	
Test Name: Application Program Interface	
Test Steps:	Comments:
1. Log onto ECS Trouble Ticket Service as User.	
2. Log into another application.	
3. Within that application, submit a trouble ticket.	
4. Enter Short Description.	
5. Enter Detailed problem description.	
6. Log-off as user.	
7. Log-on as Internal user.	
8. Query the trouble ticket that was recently submitted.	
9. Verify that the trouble ticket was submitted via another application.	C-MSS-57520
10. Send notification via E-Mail to the user.	

4.3.2 Software CM Thread (TC008)

The Software Change Manager is a software application that will provide the mechanisms to organize and partition software, control software changes, provide version control, and assemble sets of software for release purposes. The Software CM thread verifies the proper operation of the Software Change Manager Services within the Configuration Management Application Service (CMAS) by testing the capability of the Software Change Manager to maintain software libraries through version control for custom software, science software, data, documentation and report files stored in the site library. The Software Change Manager uses the COTS product, ClearCase, to perform these functions. In addition, custom scripts and triggers were developed to enable ClearCase to automatically enforce specified ECS policies regarding software management in the ECS. The policies enforce access control, modification controls, and promotion level statusing on library files according to ECS stipulated policies. Custom scripts

were also developed for frequently recurring operations. ClearCase is used as the Cmtool at bothe development and DAAC environments. In the DAAC environment there will be two sets of VOBs. One set is used for the ECS software (both source and executable code) and the other set is used for Science software and science algorithms. Since ClearCase does not differentiate science software from any other type of executable, these tests will not explicity operate using science software but any arbitrary executable to represent science algorithms and software.

4.3.2.1 Test Case 1: Verify Maintenance of Software Library Elements (TC008.001)

This test verifies that the MSS configuration management application service has the ability to maintain various types of software library files and use the Clearcase checkout/edit/checkin paradigm to govern any changes made to the library elements. Elements will be checked out and checked in to the software library as a new version of the original file.

Test Configuration:

Hardware: SUN clearcase client and server, HP client,SGI client

Software: Clearcase (COTS) and Custom Trigger scripts, authorization tables

Data: Sampling of Site Software library items

Tools: N/A

Test Input:

Inputs to this test consists of a representative sampling of software library files which include many of the following: source code, binaries and executables, patches, calibration coefficients and control data, scripts, designs and design specifications, databases, technical documentation (both text and graphics), test data, test reports, interface specifications, and configuration data.

Test Output:

Outputs of this test case consist of ClearCase messages, version control numbers and resulting logs and status messages.

Success Criteria:

This test case is successful when a representative sample of the stipulated test input is checked in to the Software change manager as elements of the software library, version numbers are appended and a log reflecting the new status of the software library contents is been produced.

Test Procedures:

Test Case ID: TC008.001	
Test Name: Verify Maintenance of Software Library Elements	
1. Test Steps:	Comments: Note: throughout this procedure "c" is an alias for the "cleartool" command
2. Log in to one of the supported architecture machines. Enter : <*login id*> on the workstation Enter: <*password*> on the workstation	This test procedure is to be performed on 3 different hardware architectures: SGI HP SUN
3. Set clearcase view and show config spec **>c setview nwoodson** > **c catcs**	 Sets the ClearCase view to "nwoodson" Displays the configuration spec used with the "nwoodson" view
4. Checkout the configured directory >**cd /ecs/SCMscience/**<*platform*>**/** >**c co -nc** .	Configured directory represents a software library. C-MSS-40480
5. Copy file elements to a configured directory >**cp /home/nwoodson/SCM/testdata/ELEMENTS/* .**	 Copies data elements: DbDeleteDriver.input DbDeleteDriver.output DpAtEnv.csh.template DpAtExecution.cxx DpAtMMgrXdiff.sh DpAtPcFileInfo.cxx DpPrExecUnitTest.plan DpPrExecutable.doc spre4sun.format.partition spre4sun.infoscript to the ClearCase configured directory C-MSS-40400
6. Verify that the files exist but are not recognized as "Clearcase" elements >**ls** >**c ls**	 Perform a unix list to verify that files do exist Perform a cleartool list to show that Clearcase does not acknowledge these as elements known to the configured directory
7. Make the data elements "ClearCase" elements. >**c mkelem -ci -nc** *	 Use the cleartool makelement command to make all the files in the configured directory "ClearCase" elements and check them back in. C-MSS-40480
8. Verify that the files in the directory are now recognized as "Clearcase" elements. > **c ls**:	 Perform a cleartool list to show that ClearCase now acknowledges that file elements as "ClearCase" elements

9. Verify by inspection that the files all have been assigned version numbers by viewing the output to the screen resulting from the “c ls” command..	C-MSS-40490
10. Checkout the configured directory > **c co** .	C-MSS-40480
11. Make the files elements in the configured directory Clearcase elements > **c mkelem -ci -nc ***	C-MSS-40480 C-MSS-40400 triggers on the mkelem command create state, promotion level and checksum attribute*s*.
12. Verify by inspection the version numbers of configured items in the software library > **c ls** **> c lshistory -long**	produces log of software library files and their version numbers. C-MSS-40490 C-MSS-40570 C-MSS-40510 C-MSS-40995 C-MSS-40990) Both requirements are partially satisfied by implementing functionality through extracted log information from standard Clearcase reports. This capability will be integrated with the proxy agent in phase 3
13. Check the configured directory back in >**c ci -nc** .	C-MSS-40480

4.3.2.2 Test Case 2: Verify the Implementation of ECS Policies on Configured File Elements (TC008.002)

This test will verify the proper implementation of ECS policies as it relates to transitioning configured file elements through software lifecycle promotion phases. These restrictions are regulated through the use of access profiles that map userids against operations the userids are authorized to perform. ClearCase does not differentiate science software from any other type of software library element. Therefore these tests use arbitrary library file elements as a representation of science software.

Test Configuration:

Hardware: SUN clearcase client and server, HP client, SGI client

Software: Clearcase (COTS),Custom software library policy scripts, authorization tables

Data: Sample Site Software library file elements to represent science algorithms

Tools: N/A

Test Input:

Inputs to this test include clearcase commands to invoke scripts that allow the user to append and change state and promotion level attributes of file elements.

Test Output:

Outputs from this test include file elements with state and promotion level attributes, custom script log messages that alert the user to the acceptance or denial of the attempts to modify file element attributes, and resulting ClearCase logs and status messages.

Success Criteria:

This test is successful when attempts to change state and promotion level attributes of file elements in the software library are allowed provided the user has been granted the correct permissions. Likewise attempts to modify the same attributes of software library elements should be denied when authorization has not been granted or when state changes within a specific promotion level are not valid

Test Procedures:

Test Case ID: TC008.002	
Test Name: Regulation of Operations on Software Library Files	
Test Steps:	Comments:
1. Login as a user other than the CM administrator Enter <*login id*> on the workstation Enter <*password* > on the workstation **>c setview nwoodson**	Log in as a user that does not have the permission to move a file element to the "inspected" state attribute. This test procedure is performed on 3 different hardware architectures: SGI HP SUN
2. Attempt to apply a state attribue to a software file element without authorizations of any type >**cd /ecs/SCMscience/**<*machine architecture*> > **/usr/ccase/scripts/change_state.sh** At the "Enter the number corresponding to the new state: " prompt, Enter: **2** Verify that the attribute "ready_for_standalone_test" was not applied to the file element by issuing the following command: >**c describe DbDeleteDriver.input**	"2" corresponds to the state attribute value of "ready_for_standalone_testing". This operation attempt should be denied because your user id is not included in any authorization table.. The following error message should be issued: "You do not have authority to perform this action." C-MSS-40470 There should be not be a state attribute value of "ready_for_standalone_testing"

3. Include your user id in one of the authorization tables. >**su -vobadm** Enter <*password* > on the workstation Edit the sdps_w authorization table and add your user id >vi /data/ccase/tables/sdps_w add your user id to the sdps_w file and save and exit	You must have pseudo root permission to modify authorization tables. C-MSS 40470
4. Attempt to apply a state attribue to a software file element with the appropriate authorizations >**cd /ecs/SCMscience/**<*machine architecture*> > /**usr/ccase/scripts/change_state.sh** **>c describe DbDeleteDriver.input** At the "Enter the number corresponding to the new state: " prompt, Enter: **2** At the "Enter a list of items to be processed." prompt, Enter: **-f DbDeleteDriver.input** At the "Are you sure you want to label the following: -f DbDeleteDriver.input for promotion to the "ready_for_standalone_test" level? [yes]" prompt , Press **RETURN** At the "Enter the value of the 'state' attribute to use as a search criteria:" prompt, Enter: **0** Verify the attribute "in_work" was applied to the DfDeleteDriver.input file element by monitoring the output to the screen or by issuing the following command: >**c describe DbDeleteDriver.input**	Invokes the script to change state attribute values. List all the existing information about the ClearCase element. Note what the state attribute value is. "2" will change the state attribute value to "ready_for_standalone_test" "-f " indicates the operation will take place on a file. DbDeleteDriver.input is the file that the attribute value will be applied to. This operation should work. The user should receive a message stating: "Created attribute "state" on DbDeleteDriver.input" C-MSS-40420 C-MSS-40470 C-MSS-40530 Check output to the screen to verify that the state attribute value is now "ready_for_standalone_test"

5. Attempt to apply a promotion level attribute to a software element	
Go to the Clearcase VOB directory >**cd /ecs/SCMscience/**<*machine-architecture*>	ie: cd /ecs/SCMscience/SUN5
>**c describe DbDeleteDriver.output**	List all attributes assigned to the file element.
Invoke the change promotion level script >**/usr/ccase/scripts/change_pl.sh**	The change_pl.sh script allows the user to apply M&O defined attributes.
At the "Enter the number corresponding to the new promotion level:" prompt, Enter: **3**	"3"corresponds to the promotion level value "standalone_test"
At the "Enter the value of the 'state' attribute to use as a search criteria:" prompt, Enter: **1**	"1" corresponds to the attribute "ready_for_standalone_test"
At the "Enter a list of items to be processed." prompt, Enter: **-f DbDeleteDriver.output**	
At the "Are you sure you want to promote to the following: -f DbDeleteDriver.output to the "standalone_test" level by search for elements with a "state" attribute of "ready_for_standalone_test"? [yes}" prompt, Press **RETURN**	Look for a status message stating "created attribute "promotionLevel" on DbDeleteDriver.output" and "created attribute "state" on DbDeleteDriver.output" C-MSS-40420 C-MSS-40470 C-MSS-40530
At the "Enter the value of the 'state' attribute to use as a search criteria:" prompt, Enter: **0**	"0" corresponds to quit
Verify the state and promotion level attributes were updated by viewing the output to the screen or by issuing the following command: >**c describe DbDeleteDriver.output**	state attribute value should = "in_standalone_testing PromotionLevel attribute value should = "standalone_test"
6. Create user defineable attributes	
>**c co DpPrExecutable.doc**	Check out the file element.
>**c describe DpPrExecutable.doc**	List all existing information about the ClearCase element.
>**c mkattr Status '"start"' DpPrExecutable.doc**	Create an attribute titled "Status" and apply the value of "start" to it.
>**c describe DpPrExecutable.doc**	List all the existing information about the ClearCase element. Verify that the new attribute "Status" is listed by viewing the output to the screen. C-MSS-40410

7. Attempt to apply an attribute with authorizations to low level transitions but no authorization to higher level transitions. Invoke the change promotion level script >**/usr/ccase/scripts/change_pl.sh** At the "Enter the number corresponding to the new promotion level:" prompt, Enter: **4**	ie: user id is included in the $SDPS_W authorization table but not included in the $SSI_T authorization table. "4" corresponds to the promotion level attribute value of "received_by_DAAC" This operation attempt should be denied because your user id is not included in the correct authorization table to transition to promotion level "received_by_DAAC". The following error message should be issued: "You do not have authority to perform this action." C-MSS-40420 C-MSS-40470 C-MSS-40530
8. Exit ClearCase >**exit** Exit process >**exit**	

4.3.2.3 Test Case 3: Verify Software Build Procedures (TC008.003)

This test verifies the capability to create a build, label the components of the build, create a branch, modify a file element, merge the edited library file with the existing one and identify version conflicts, if necessary. Edited file elements are saved as a new version of the original file.

Test Configuration:

Hardware: SUN clearcase client and server, HP client, SGI client

Software: Clearcase (COTS),Custom software library policy scripts, authorization tables

Data: Sample Site Software library file elements that comprise a build

Tools: N/A

Test Input:

Inputs to this test include file elements necessary to comprise a build and ClearCase commands required to manipulate the configured file elements.

Test Output:

Outputs of this test case consist of: version control numbers, attributes, labels appended to file elements and resulting ClearCase logs and status messages.

Success Criteria:

This test is successful when a build is successfully performed, file elements that comprise that build are given appropriate attributes and labels, a successful merge is performed of different versions of an element and version conflicts as a result of the merge are acknowledged and resolved.

Test Procedures:

Test Case ID: TC008.003	
Test Name: Verify Software Build Procedures	
Test Steps:	Comments:
1. Log in to one of the supported architecture machines. Enter : <*login id*> on the workstation Enter: <*password*> on the workstation	This test procedure is performed on 3 different hardware architectures: SGI HP SUN
2. Set clearcase view >**c setview nwoodson**	
3. Set clearcase config spec and display it. >**c setcs -default** >**c catcs**	Set the configuration spec to the default Display what the current config spec is. Current config spec should be: element * CHECKEDOUT element: * /main/LATEST
4. Go to the Clearcase VOB directory > **cd /ecs/SCMscience**	
5. Go to directory of the host plaform >**cd** <*platform*>	ie: cd SUN5 cd HP cd SGI
6. Check out the platform directory >**c co -nc** .	Checks out the platform directory so that a BUILD directory can be created below it.
7. Make a clearcase directory named "BUILD" to keep build elements >**c mkdir -nc BUILD**	Creates a ClearCase configured directory.
8. Check in the platform directory >**c ci -nc** .	Checking in the platform directory allows pubic to see plaform directory file elements.
9. Change your cwd to BUILD >**cd BUILD**	
10. Copy all build elements to the BUILD directory >**cp /home/nwoodson/SCM/testdata/ELEMENTS/*time* .**	Copies : make.time make file time3.cxx source file time3.h include file time3drv.cxx driver build elements to the ClearCase configured directory.

11. Make all build files clearcase elements and check them in >**c mkelem -ci -nc ***

24. Attach the label “BUILD1” to all build elements >**c mklabel -config timedrv**_<platform>.**tc1 BUILD1**	
25. Verify that the “BUILD1” label was applied to all file elements in the build >**c describe * >** <platform>_**build1.label**	
26. Do a unix list and a ClearCase list >**ls** >**c ls**	
27. Display current config spec >**c catcs**	
28. Change the config spec >**c setcs /home/nwoodson/SCM/<platform>_bug_view**	Change the config spec to enable the editing of a file in the build as a bug fix and merge it back to the <platform>_build.
29. Display the new config spec >**c catcs** >**c ls**	Tthe config spec should be changed to: element * CHECKEDOUT element * /main/<platform>_build/bug_fix/LATEST element * /main/<platform>_build/LATEST -mkbranch bug_fix element * /main/LATEST -mkbranch sun_build
30. Check out a file element >**c co -nc time3.cxx**	Checks out time3.cxx on the bug_fix branch
31. Verify by inspection that the file element is checked out on the bug_fix branch >**c ls**	
32. Copy an edited version of a file element (in the build) over the existing copy of the file element >**cp /home/nwoodson/tme3.cxx.temp time3.cxx**	Simulates editing a version of a file element
33. Check the new file back in to ClearCase >**c ci -nc time3.cxx** >**c ls**	Creates a new version of the time3.cxx file and assigns version numbers C-MSS -40490
34. Build the executable and do not create configuration records. >**clearmake -C gnu -F -f make.time** >**mv timedrv timedrv**_<*platform*>**.tc2.nocr**	
35. Remove all object files >**rm *.o**	
36. Rebuild executable and create configuration records >**clearmake -C gnu -f make.time** >**mv timedrv timedrv**_<*platform*>**.tc2**	
37. Display the history and audit trail of the build elements and store it in a file >**c catcr -long -r timedrv**_<*platform*>**.tc2 >**<*platform*>_**build2.audit**	C-MSS-40510 C-MSS-40570
38. Return to the previous view >**c setcs /home/nwoodson/SCM**/<*platform*>_**view** **>c catcs** **>c ls**	 Display the new view Display all the “ClearCase:” elements

39. Merge executable from the bug_fix branch to the <platform>_build branch >**c co -nc time3.cxx** >**c merge -to time3.cxx -version /main/<***platform***>_build/bug_fix/LATEST** **>c ls**	Check out time3.cxx to merge to the <platform>_build branch Merges time3.cxx from bug_fix branch to <platform>_build branch C-MSS-40500
40. Check the merged version of the executable back in >**c ci -nc time3.cxx** >**c ls**	
41. Verify by inspection that file labels, file history and merge history are maintained on the modified build element. >**c lshistory time3.cxx**	C-MSS-40510 C-MSS-40570 C-MSS-40990 C-MSS-40995
42. Build the executable once again without config records based on the merged version of the executable >**clearmake -C gnu -F -f make.time** **>mv timedrv timedrv_<***platform***>.tc3.nocr** **>rm *.o**	
43. Rebuild the executable and generate config records based on the merged version of the executable >**clearmake -C gnu -f make.time** **> mv timedrv timedrv_<***platform***>.tc3**	
44. Display the history and audit trail of the build elements and store it in a file >**c catcr -long -r timedrv_<***platform***>.tc3 >**<platform>**_build3.audit** **>c ls**	

4.3.2.4 Test Case 4: Verify Ability to Recreate a Build (TC008.004)

This test verifies that the MSS configuration management application service has the ability to recreate a build. This test uses files and setup previously generated in test case number TC008.003.

Test Configuration:

Hardware: SUN clearcase client and server, HP client, SGI client

Software: Clearcase (COTS) and Custom Trigger scripts, authorization tables

Data: Sample Site Software library file elements that comprise the build generated in test case 3

Tools: N/A

Test Input:

Inputs to this test consists of file elements necessary to comprise the build generated in test case TC008.003 and ClearCase commands required to manipulate the configured file elements.

Test Output:

Outputs of this test case consist of an exact reproduction of the executable produced in test case 3.

Success Criteria:

This test case is successful when the ability to recreate a previous build is demonstrated. The rebuild capability is facilitated by labels appended at the time of the intial build.

Test Procedures:

Test Case ID: TC008.004	
Test Name: Verify Ability to Recreate a Build	
Test Steps:	Comments: Note: throughout this procedure "c" is an alias for the "cleartool" command
1. Set your config spec to point to all build elements that had the BUILD1 label attached to them in testcase3. **>c catcs /home/nwoodson/SCM/<platform>_BUILD1_view** **>c catcs** **>c ls**	Display all the build elements that are labeled as BUILD1
2. Repeat the build from the build elements that are labeled as BUILD1 and do not create configuration records. **>clearmake -C gnu -f make.time** **>mv timedrv timedrv_<platform>.tc4.nocr** **>rm *.o**	
3. Repeat the build from the build elements that are labeled as BUILD1 and create configuration records. **>clearmake -C gnu -f make.time** **>mv timedrv timedrv_<platform>.tc4**	C-MSS-40550
4. Display history and audit trail of build elements **>c catcr -long -r timedrv_**<*platform*>**.tc4 >** <*platform*>**_build4.audit** **>c ls**	C-MSS-40990 C-MSS-40995
5. Verify that the two builds each used a different config spec **>diff** <*platform*>**_build3.audit** <*platform*>**_build4.audit > aud3_aud4_**<*platform*>**.diff**	This verifies that each build had a different configuration record.

4.3.2.5 Test Case 5: Verify the Capability For Concurrent Use of Library File Elements (TC008.005)

This test verifies that the MSS configuration management application service allows users the ability to work concurrently on the same library file element and identify version conflicts.

Test Configuration:

Hardware: SUN clearcase client and server, HP client, SGI client

Software: Clearcase (COTS) and Custom Trigger scripts, authorization tables

Data: Sample Site Software library items

Tools: N/A

Test Input:

Inputs to this test consists of ClearCase commands to operate on a specific Library file element.

Test Output:

Outputs of this test case consist of ClearCase status messages and logs.

Success Criteria:

This test case is successful when two users are able to modify library file elements concurrently.

Test Procedures:

Test Case ID: TC008.005	
Test Name: Verify the Capability For Concurrent Use of Library file Elements	
Test Steps:	Comments: Note: throughout this procedure "c" is an alias for the "cleartool" command
1. User 1 login to workstation Enter: <*login-id*> on the workstation Enter <*password*> on the workstation	
2. 2.Set ClearCase view to nwoodson_<*platform*> >**c setview nwoodson-<***platform***>**	
3. Show current config spec >**c catcs**	
4. Set config spec to to <*platform*>_view >**c setcs /home/nwoodson/SCM/<***platform***>_**view	
5. Show new config spec > **catcs**	
6. Go to the ClearCase configured directory >**cd /ecs/SCMscience/**<*platform*> **/BUILD**	
7. Produce a UNIX list of the file elements >**ls**	
8. Produce a ClearCase list of file elements >**c ls**	
9. User 2 login to workstation Enter: <*login-id*> on the workstation Enter <*password*> on the workstation	
10. Set ClearCase view to nwoodson >**c setview nwoodson**	
11. Show current config spec >**c catcs**	
12. Set config spec to to <*platform*>_view >**c setcs /home/nwoodson/SCM/<*platform*>_view**	

13. Show new config spec > **catcs**	
14. Go to the ClearCase configured directory >**cd /ecs/SCMscience/**<*platform*>**/BUILD**	
15. Produce a UNIX list of the file elements >**ls**	
16. Checkout a file element >**c co -nc time3.cxx**	
17. Edit the file >**vi time3.cxx** insert comment in file **// lines added for** <*platform*> **concurrent test** **int i =i+10;** **exit time3.cxx file**	
18. Check in the new version of the file element >**c ci 0nc time3.cxx**	
19. Produce a ClearCase list >**c ls**	
20. User 2 Attempt to check out the same file elementl >**c co -nc time3.cxx.**	Should receive a Cleartool error message.
21. Verify that User 2 receives a ClearCase error message by viewing the output to the screen	User 2 should receive the following error message: cleartool: Error: Branch "/main/hp_build" of elemetn is checked out reserved by view "msse3sun:/data/viewstore/nwoodson.vws". Unable to check out "time3.cxx"
22. User 2 Check the file element out in the unreserved mode. >**c co -unreserv -nc time3.cxx**	ClearCase should allow this acitvity.
23. User 2 Edit the file >**vi time3.cxx** insert comment in file **// lines added for** <*platform*> **concurrent test** **i =i+5;** **int i:;** **exit time3.cxx file**	Now both users can modify the same version of the file concurrently. ClearCase stipulates the user holding the unreserved copy of the file must wait until the user of the reserved copy checks their version in. C-MSS-40560
24. User 2 Attempt to check in the unreserved version of the file element. >**c ci -nc time3.cxx**	ClearCase should identify that: The most recent version of the existing file element is not the version that the unreserverd user was modifying.
25. Verify that ClearCase issues an error message	ClearCase produces the following error message: Cleartool: Error: The most recent version on branch "/main/hp_build" is not the predecessor of this version. Cleartool: Error: Unable check in "time3.cxx"
26. User 2: Resolve the conflict by performing a merge of the two version of the file eleement >**c merge -to time3.cxx -version /main/hp_build/2**	Merge the User 2 copy of time3.cxx with the most recent version of time3.cxx in ClearCase now.

27. User 2: Verify that ClearCase merges the two file versions by examining the output to the screen	Expected ClearCase messages should show the differences in the two files and how it proposes to merge the two files. ClearCase also issues messages indicating that the contributor file is moved to <filename>.contrib, Name of the file that the output is written to and name of the file that the merge is recorded to. C-MSS-40500
28. User 2: Check in the merged version of the file element >**c ci -nc time3.cxx**	time3.cxx should be checked in as a new version of the previous file.
29. User 1 /User 2 View the results of the merge >**more time3.cxx**	

4.3.3 Physical Configuration Management Thread (TC009)

The Physical CM Service provides the capability to track, manage, and control all the physical elements in the configuration. It integrates graphics with data to create a complete electronic model of the physical infrastructure of the configuration. It supports a variety of configuration administration applications including inventory, billing, and troubleshooting.

The Physical CM Service is provided by the COTS package Physical Network Manager/MountainView by Accugraph. The Physical Network Manager/MountainView package is responsible for transforming the logical configuration management environment into a physical one. It has a graphics module to create new drawings or to import existing facility drawings to create a physical layout of the configuration.

The Physical CM thread consists of testing the installation of the products and testing the sharing of the managed objects with HpOpenView. The COTS package Physical Network Manager/MountainView will also be tested for its ability to satisfy its requirements.

4.3.3.1 Test Case 1: Test the Physical CM GUI (PNM) (TC009.001)

This test case verifies the main features of the Physical CM COTS tool. It includes graphical interface for viewing the physical location of system components on the floor plans and changing the location of the system components.

Test Configuration:

Hardware: MSS Server (MSSE4HP)

Software: Physical Network Manager/Mountain View (Physical CM COTS package), Sybase

Data: Hardware and Software Resource Objects

Tools: HpOpen View (Physical location of resources)

Test Input:

Operator actions to exercise the capabilities of the GUI.

Test Output:

Displays and printouts resulting from operator actions.

Success Criteria:

This test is successful if GUI-specific actions and pull-down screens correctly respond to user input.

Test Procedures:

Test Case ID: TC009.001	
Test Name: Test the Physical CM GUI	
Test Steps:	Comments:
	CONVENTIONS USED IN THIS PROCEDURE: Bold face type - the specific text you are to type is shown in bold face. Menu selections are indicated in boldface type and are preceded with the word "Select". Place holders - place holders for items such as filenames that you must supply are italicized and enclosed in braces. ie:<*filename*> Keys - all keys on the keyboard that are used will be shown using small capital letters. ie: the Control Key = CNTRL, Return key = RET Mouse button keys - mouse button keys are indicated as follows: LMB - Left mouse button MMB - Middle mouse button RMB - Right mouse button
1. Invoke the Physical CM COTS GUI (PNM). > **cd /users/mtp/prog** **>./pnm &**	Puts you in the PNM directory Invokes the PNM COTS tool The Accugraph-Mountain Top window will appear. This verifies access to the Physical Network Manager.
2. Login SYBASE database	
2a. Click RMB to bring up the Mtp-Main Menu	The Mtp-Main Menu window will appear.

2b. From the Mtp-Main Menu select **TOOLS>RISE> DB DEFS>DB ACCT DEFS>DB ACCT LOGIN> SYBASE**	This gives PNM access to the SYBASE database. Input line <rep> window will appear.
2c. Respond to the prompts with the following inputs**:** Enter account name: **mtp_role** password: **<***enter your password***>** Database Name: **mtpdb** Database Server Name: **msse4hp_srvr** Do you wish to login to database: Select **OK**.	
2d. Click the MMB to remove the Mtp-Main Menu	
3. View the physical floor plan location.	C-MSS-87560
3a. From the PNM Top-Menu select **Load File**	The Mtp-Main Menu window will appear.
3b. Enter **/home/<**username**>/pnm/newdraw/usa** in the input line window <RET>	The selected drawing will appear. The available drawings in our test configration are: a) USA b) EDF Daac c) Rack 1 d) Rack 2 e) Rack 3
3c. Double click on the dot in the drawing that represents Maryland.	Pop up window will appear.
3d. Respond to the following prompt in the pop up window: Do you want to replace this file:<*filename*>?Select **No**	The Mtp-Main Menu window will appear.
3e. Click MMB to remove the Mtp-Main Menu.	The USA physical map unfolds to show the Maryland Map.
3f. Double click on the dot in the Maryland map that represents HITC.	Pop up window will appear.
3g. Respond to the following prompt in the pop up window: Do you want to replace this file:<*filename*>?Select **No**	The Mtp-Main Menu window will appear.
3h. Click MMB to remove Mtp-Main Menu.	Mtp-Main Menu window will disappear and the HITC map unfolds to display the EDF Daac map.
3i. Double click on the dot that represents Rack 1,Rack 2 or Rack 3.	Pop up window will appear.
3j. Respond to the following prompt in the pop up window: Do you want to replace this file:<*filename*>?Select **No**	The Mtp-Main Menu window will appear.
3k. Click MMB to remove Mtp-Main Menu.	Mtp-Main Menu window will disappear and the EDF Daac map unfolds to display the select rack map.
3l. Select from Top-Menu **Designer**, from the Sub-Menu select **View Object**	Allows you to view object information.
3m. Select the object whose attributes you would like to display.	Database Columns window will appear which displays all attributes related to the object you choose.
3n. Select cancel from the Database Columns window to exit.	
3o. Select softkeys button to bring up softkeys menu.	.
3p. Select Redraw from softkeys menu.	To redraw the map displayed
4. Change the physical location of a system component.	C-MSS-87570

4a. Select from Top-Menu **Graphic Input**, from the Sub-Menu select **Draw>Move**	Allows you into the drawing environment.
4b.Click on the object you'd like to move, drag and drop the object to the desired location and press <RET> or the LMB.	Allows you to move the physical location of a selected component.
4c. Exit the drawing. Select **Exit Draw**	Returns you to your previous operation.
5. Add data items to devices on the system floor plan.	C-MSS-87510
5a. Select from Top-Menu **Designer**, from the Sub-Menu select **Add Object>Devices**	Allows you to select a pre-defined (PNM) or a user-defined object and incorporate it into your drawing.
5b.Choose your pre-defined device type in window box. Click LMB (your selection will be hi-lighted).	Window appears to choose a device type for placement.
5c. Select **Accept** in the window box.	Note: Each Daac site may incorporate objects such as cables, devices and menu graphics.
6. Verify the data items were added.	C-MSS-87630 C-MSS-87640
6a. Select from Top-Menu **Designer**, from the Sub-Menu select **View Object**	Allows you to view information about the selected component.
6b. Select the object whose attributes you would like to verify.	Database Columns window will appear to which displays all attributes related to the object you choose.
6c. Select cancel from the Database Columns window to exit.	
6d. Select softkeys button to bring up softkeys menu.	To redraw the map displayed.
6e. Select Redraw from softkeys menu.	
7. Modify the items on the system floor plan.	C-MSS-87510
7a. Select from the Top-Menu **Designer**, from the Sub-Menu select **View Object**	Allows you to view information about the selected component.
7b. Select the object whose attributes you would like to modify.	Database Columns window will appear to which displays all attributes related to the object you choose.
7c. Enter the modifications in window box	Allows you to modify parameters on the object.
7d. Select **Accept** twice in both window boxes.	Accepts all your modifications.
8. Verify the data items were modified.	C-MSS-87630 C-MSS-87640
8a. Select from the Top-Menu **Designer**, from the Sub-Menu select **View Object**	Allows you to view information about the selected component.
8b. Select the object whose attributes you would like to verify	Database Columns window will appear to which displays all attributes related to the object you choose.
8c. Select cancel from the Database Columns window to exit.	
8d. Select softkeys button to bring up softkeys menu.	To redraw the map displayed.
8e. Select Redraw from softkeys menu.	

4.3.3.2 Test Case 2: Generate Physical CM Reports (TC009.002)

This test case verifies the capability to generate standard and customizable reports of Physical CM data using the PNM Physical CM COTS tool.

Test Configuration:

Hardware: MSS Server (MSSE4HP)

Software: Physical Network Manager/MountainView (Physical CM COTS package), Sybase

Data: Hardware and Software Resource Objects

Tools: HpOpen View (Physical location of resources)

Test Input:

Operator actions to generate Physical CM reports.

Test Output:

Physical CM reports generated by the COTS product.

Success Criteria:

This test is successful when Physical CM reports can be created, accessed, and verified to contain the proper information.

Test Procedures:

Test Case ID: TC009.002	
Test Name: Generate Physical CM Reports	
Test Steps:	Comments:
1. Generate a standard report(s) in the configuration items database.	C-MSS-87600
1a. Select **Load File** from Top-Menu option	Load Mtp Menu window will appear.
1b.Enter **/home/<**username**>/pnm/newdraw/<Daac>** in the input line window <RET>	The selected drawing will appear.
1c. Click RMB to bring up the Mtp-Main Menu.	The Mtp-Main Menu window will appear.
1d. Select **TOOLS>RISE>DB ADD> MODE:FORM>ADD ENTRY RECORD**	Mode change allows the user to switch from “Normal” to “Form” mode for report generation.
1e. Click MMB to close Mtp-Main Menu window.	
1f. Select the desired object and say yes.	The Database table window will appear.
2. Selection of various report(s) (tables).	C-MSS-87610

2a. Select a table (report).	Allows you the user to select the data information on the related object. The following reports are available in PNM: **MTPH2-Software**, inventory data (item name, purchase date, purchase price, installation date, manufacturer, serial #, physical location) **MTPH3-Network**, network data (network location, protocols) **MTPH1-Maintenance**, maintenance data (date maintenance performed)
2b. Click LMB and the status information on object appears.	
2c. Select on softkeys **EXIT Editor** from table.	Withdraws you from table.
2d. Are you sure you wish to add this entry? Select **NO**	Window box appears to add an entry to to table (report).
2e. Click MMB to remove the Add Entry window box.	The Mtp-Main Menu window will appear.
3. Customize data report(s) by modifying information.	C-MSS-87610
3a. Select **Load File** from Top-Menu option	Load Mtp Menu box will appear.
3b.Enter **/home/<username>/pnm/newdraw/<Daac>** in the input line window <RET>	The selected drawing will appear.
3c. Click RMB to bring up the Mtp-Main Menu.	The Mtp-Main Menu window will appear.
3d. Select **TOOLS>RISE>DB ADD>** **MODE:FORM>ADD ENTRY RECORD**	Mode change allows the user to switch from “Normal” to “Form” mode for report generation.
3e. Click MMB to close Mtp-Main Menu window.	
3f. Select the desired object and say yes.	The Database table window will appear.
3g. Select a table (report).	Allows the user to select the data information on related objects. The following reports are available in PNM: **MTPH2-Software**, inventory data (item name, purchase date, purchase price, installation date, manufacturer, serial #, physical location) **MTPH3-Network**, network data (network location, protocols) **MTPH1-Maintenance**, maintenance data (date maintenance performed)
3h. Click LMB and the status information on object appears.	Window box appears to add an entry to to table (report).
3i. Are you sure you wish to add this entry? Select **YES**	Allows user to edit information associated with the report environment.
3j. Enter in item information.	
3k. Select on softkeys **EXIT Editor** from table.	Exits table.

4.3.3.3 Test Case 3: Importing an Existing floor plan (TC009.003)

This test case verifies the actions associated with the importing and storing of an existing system floor plan. Associated reports are produced for verification. All physical system components

such as inventory data, network data and maintenance data are maintained. The capability of adding and modifying information regarding physical components that cannot be discerned through ECS management framework is verified.

Test Configuration:

Hardware: MSS Server (MSSE4HP)

Software: Physical Network Manager/MountainView (Physical CM COTS package), Sybase

Data: Existing Scanned Image file, Hardware and Software Resource objects

Tools: HpOpen View (Physical location of resources)

Test Input:

Operator actions to enter data item information into the CM tool.

Test Output:

Reports listing data items entered.

Success Criteria:

This test is successful when Physical CM reports accurately reflect the information in the CM database.

Test Procedures:

Test Case ID: TC009.003	
Test Name: Maintain Physical CM Records	
Test Steps:	Comments:
1. Import an existing floor plan.	C-MSS-87500
1a. Click RMB to bring up the Mtp Main Menu.	The Mtp-Main Menu window will appear.
1b. Select **TOOLS>SCAN>FILE>LOAD IMAGE**	The systems displays the default directory path in the input window and its files or subdirectories in the menu.
1c. Enter <*filename*> in window box.	Imported scanned image drawing will appear in small scale.
1d. Select **EXIT>DISPLAY>FULL VIEW**	Full scale drawing will be displayed.
1e. Click RMB to bring up Mtp-Main Menu box and exit image.	The Mtp-Main Menu window will appear.
1f. Select **TOOLS>SCAN>EDIT>EXIT DRAW**	
2. Add physical components to imported floor plan.	C-MSS-87550
2a. Select from Top-Menu **Designer**, from Sub-Menu select **Add Object>Devices**	Allows you in the network mode for interfacing with network objects.
2b. Select in box <pick workstation>.	Window will appear for devices selection.
2c. Select **Accept** in window box, **size, place drawing**, and **save** the file.	Updates status in the database.
2d.Enter **/home/<username>/pnm/draw/<***filename***>**	The Mtp-Main Menu window will appear.

2e. Select from Top-Menu **Load File**, enter your **<***filename***>.**	Load Mtp Menu box will appear.
2f. Attach and select the object and yes with LMB.	
2g. Select **Exit>MTPT1** (table), select **yes** with LMB.	Selection of database table.
2h.Select from Top-Menu **Designer**, from the Sub-Menu select **View Object**	Allows you to view information about the selected component.
2i. Select object and enter information.	
3. Store the component information.	C-MSS-87530
3a. Select **Accept** twice in window boxes.	Updates status in the database.
3b. Select from Top-Menu **Network Manager**, from the Sub-Menu select **Update Status>One Node**	Allows you to update the network status for a selected single node.
4. Modify fields after storage of physical components.	C-MSS-87550
4a. Select from Top-Menu **Designer**, from the Sub-Menu select **View Object**	Allows you to view information about the selected component.
4b. Select the object whose attributes you would like to modify.	Database Columns window will appear, which displays all attributes related to the object you choose.
4c. Enter the modifications in the window box.	Allows you to modify parameters on the object.
4d. Select **Accept** in window boxes.	Updates status in the database.
5. Load information for ECS components to reflect inventory, network and maintenance.	C-MSS-87580
5a. Click RMB to bring up the Mtp-Main Menu.	The Mtp-Main Menu window will appear.
5b. Select **TOOLS>RISE>DB ADD>ADD ENTRY RECORD**	Allows you to add operations pertaining to components.
5c. Select object and yes with LMB.	The Database table window will appear.
5d. Select table type and enter **<***information***>.**	
5e. Exit the table on softkeys **Exit Editor**.	
6. Search and retrieve information from the tables via PNM SQL search language.	C-MSS-87590
6a. Click RMB to bring up the Mtp-Main Menu.	The Mtp-Main Menu window will appear.
6b. Select **TOOLS>RISE>DB SEARCH> EZY QUERY**	The Database table window will appear.
6c. Select the SQL table and click LMB.	Brings up SQL database information.
6d. Click LMB to exit out of table.	
7. Generate a standard set of reports against this data.	C-MSS-87600
7a. Select **Load File** from Top-Menu option	Load Mtp Menu box will appear.
7b.Enter **/home/<username>/pnm/newdraw/<Daac>** in the input line window <RET>.	The selected drawing will appear.
7c. Click RMB to bring up the Mtp-Main Menu.	The Mtp-Main Menu window will appear.
7d. Select **TOOLS>RISE>DB ADD> MODE:FORM>ADD ENTRY RECORD**	Mode change allows the user to switch from “Normal” to “Form” mode for report generation.
7e. Click MMB to close Mtp-Main Menu window.	
7f. Select the desire object and say yes.	The Database table window will appear.
7g.Select table (report) click LMB.	
7h.Select on softkeys **EXIT Editor** from table.	Withdraws you from the table.
8. Generate a custom set of reports against this data.	C-MSS-87610
8a. Select **Load File** from Top-Menu option	Load Mtp Menu box will appear.
8b.Enter **/home/<username>/pnm/newdraw/<Daac>** in the input line window <RET>.	The selected drawing will appear.
8c. Click RMB to bring up the Mtp-Main Menu.	The Mtp-Main Menu window will appear.

8d. Select **TOOLS>RISE>DB ADD> MODE:FORM>ADD ENTRY RECORD**	Mode change allows to switch from Normal to Form mode for report generation.
8e. Click MMB to close Mtp-Main Menu window.	
8f. Select the desire object and say yes.	The Database table window will appear.
8g.Click LMB and the status information on object appears.	Window box appears to add an entry to to table (report).
8h. Are you sure you wish to add this entry? Select **YES**	Allows you editing information associated with the report environment.
8i. Enter in item <*information*>.	
8j. Select on softkeys **EXIT Editor** from table.	Exits table.

4.3.3.4 Test Case 4: Modifying a Node (TC009.004)

This test case verifies the actions associated with adding a new node to the system, moving an existing node to a new location in the system, checking the nodal attributes in the database and deleting an existing node from the system.

Test Configuration:

Hardware: MSS Server(MSSE4HP)

Software: ECS Manager Framework (HpOpenView), Physical Configuration Manager, Network Manager, Sybase

Data: Hardware and Software Resource Objects

Tools: HpOpen View (Physical location of resources)

Test Input:

Operator actions to add a node to the network.

Test Output:

Notification that a node has been added to the network.

Success Criteria:

This test is successful when a new node is added to the network, nodal attributes are automatically gathered and added to the database, additional information can be added manually, and the physical map is updated to include the new node symbol. The information concerning the new location of the node is configured into the system. Deletion of a node from the system and that the node symbol on the network map is removed with all its database links deleted.

Test Procedures:

Test Case ID: TC009.004	
Test Name: Modifying a Node	
Test Steps:	Comments:
1. Configure network software into Physical CM system.	The Network Manager Software will automatically add attribute data associated with the node to the database.
2. Add additional information regarding nodal attributes.	C-MSS-87510 C-MSS-87540
2a. From Top-Menu Select **Designer**, from the Sub-Menu select **View Object**	Allows you to view information about the selected component.
2b. Select the object and enter **<***information***>**.	Window box will appear.
2c. Select **Accept** twice in window boxes.	Updates status in database.
3. Verify the network physical map is updated to include the new node object status.	C-MSS-87520
3a. Select from Top-Menu **Network Manager**, from the Sub-Menu select **Update Status>One Node**	Allows you to update the network status for a selected single node.
3b. Select the object and click LMB.	Displays symbol as updated.
4. Determined the status of database node object with the appropriate information fields.	C-MSS-87530
4a. From the Top-Menu Select **Network Manager**, from the Sub-Menu select **Update Status>Drawing Nodes**	Allows you to update the network status for nodes within the current drawing.
4b. Select **Exit** to exit status.	
5. Move an existing node object. Display the new location when prompted.	C-MSS-87510
5a. Select from Top-Menu **Asset Manager**, from the Sub-Menu select **Manager M-A-C>Move>Move Across**	Allows you to move an object within the database.
5b. Select **Accept** in the window box.	Updates status in database.
5c. Drag and drop object to the desired location.	The Mtp-Main Menu window will appear.
5d. Mtp-Main Menu pops up click #0.	
5e. Select **location**, **designation** and **identify object** and **next drawing**.	Allows for sizing of the object.
5f. Size the drawing by pressing **/**	Allows you to enlarge the object.
6. Verify the move by checking the nodal attributes in the database and that all its links are preserved.	The Network Manager will send a 'move node' message to the Physical Configuration Manager and the move will be handled automatically.
6a. Select from the Top-Menu **Network Manager**, from Sub-Menu select **Update Status>One Node**	Allows you to update the network status for a selected single node.
6b Select the object, click LMB.	
7. Delete a node from the system.	C-MSS-87510 Note: Processing will be handled automatically by the Physical Configuration Manager.
7a. From the Top-Menu Select **Designer,** from the Sub-Menu select **Remove Object**	Allows you to removed an object and its associated database information.
7b. Select the object and Select **YES.**	
7c. Select the choice to delete in window box to remove the object from database.	The Database window box appears for your selection.
7d. Select cancel to exit window.	

8. Verify attribute data associated with the node object is removed from the database with all its links deleted.	C-MSS-87590
8a. Click RMB to bring up the Mtp- Main Menu.	The Mtp-Main Menu window will appear.
8b. Select **TOOLS>RISE>DB SEARCH> EZY QUERY**	The Database table window will appear.
8c. Select a any table to check that status as been removed. Click LMB.	The Database information is void.
8d.Select on softkeys **Exit Editor** to exit from table.	

4.4 Communications Infrastructure 1 Tests

The following threads and builds verify the functionality of asynchronous message passing and directory naming in a basic communications environment.

- Message Passing 1 Thread (TC010)
- Directory Naming Thread (TC011)
- Communication Infrastructure 1 Build (BC003)

4.4.1 Message Passing One Thread (TC010)

The purpose of this thread is to verify the functionality of implementation one of the Message Passing Service. Implementation one will provide for asynchronous and synchronous message passing with recovery-persistence. The concept of "groups" is included in this API. A group is a symbolic name representing a number of receivers. Each receiver will be identified by a logical name associated with an UUID. This API will handle messages consisting of ascii streams, binary streams, and objects. This implementation provides mechanisms for developers to utilize both asynchronous and synchronous communication in their applications. These methods of communication will be utilized in a multi-platform DCE environment consisting of multiple instances of various client and server applications.

Asynchronous message passing mode occurs when the client makes a non-blocking request to the server; i.e. without loosing interactive control. The request(call) will not return anything and is just used to pass data to a server. This allows client execution to continue while the server processes the received data.

Synchronous message passing mode occurs when the client makes a blocking request(call) and passes control to the server. The server processes the request and returns the result back to the client, at which point the client gets back control.

4.4.1.1 Test Case 1: Guaranteed Asynchronous Message Passing (TC010.001)

The purpose of this test is to verify that implementation one of the Message Passing Service has the capability to transmit messages/data via the asynchronous message passage mode of communication and guarantee the successful completion of the transmission. Specifically, this test demonstrates that implementation two of the Message Passing Service:

- provides a non blocking API for clients(senders) to send messages to servers(receivers) asynchronously.
- supports guaranteed delivery of the message to the receiver.
- can associate the receiver and the message object to a return value that is available upon request to the sender.
- provides an API for senders to obtain return information concerning the transmission of the message.

Test Configuration:

Hardware: HP and Sun workstations/servers

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions

Data: Tester supplied input

Tools: guar_cli_hp, srv_hp, guar_cli_sun, srv_sun.

Test Input:

The name under which the test server application (e.g. srv_sun) will be registered in the cell directory service must be supplied to both the test server and client applications.

Test Output:

The client test driver should issue statements indicating that it is initializing the message service, creating a callback receiver and its associated queue, along with a callback object. It will then create a sending session list, add the callback receiver to the session list, send the message and it continue its execution. It should then display the uuid of the message object, the uuid of the receiver, an acknowledgment that the message was delivered to the intended receiver, and the return status of the message transmission. The server test driver should issue a statement indicating that the message was transferred.

Success Criteria:

The client application sends a message to the server and continues executing and subsequently receives an acknowledgment indicating that the message was transmitted successfully and the message was retrieved from the queue.

Test Procedures:

Test Case ID: TC010.001	
Test Name: Guaranteed Asynchronous Messaging Passing	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.
2. Create two sessions (i.e. open two xterm windows) and log into DCE in both by typing : **dce_login** *username password*	
3. In one window, start the cdsbrowser application in the background by typing: **cdsbrowser &**	Note the cdsbrowser application is available only on HP platforms. When executing this testcase on other platforms create a third session in step 2. In this window log into a HP machine in the same cell as the test machine, set your DISPLAY variable to point back to the test machine, and then execute step 3.
4. Start the test server application by typing the following in one of the test machine windows: **srv_hp** *<cds entry name>*	The cds entry should be placed in a directory in which the executor of the test has write permission. For example, in the relacell the following is a valid entry name: */.:/RelA/msg_server* where */.:/RelA/* is the directory and *msg_server* is the server name. A message stating "The server is now listening...." should appear in the server window indicating that the server is running.
5. Verify the server object has been registered in the CDS by clicking the **tree->update** button in the cdsbrowser's toolbar.	The server object should now be displayed inside the user-specified directory (e.g. underneath the */.:/RelA/* directory)
6. In the second window start the client application by typing the following: **cli_hp** *<cds entry name>*	The cds entry name should correspond to the cds entry name specified in step 4.
7. Verify a message indicating that the Message Passing service has been initialized appears in the client window.	
8. Verify a message indicating that a callback object has been created appears in the client window.	
9. Verify a message indicating that a callback receiver has been created appears in the client window.	
10. Verify a message indicating that a session list has been created appears in the client window.	
11. Verify a message indicating that the callback receiver has been added to the session list appears in the client window.	
12. Verify a message indicating that the message has been sent appears in the client window.	C-CSS-22090

13. Verify a message indicating the client has regained control of the process's execution appears in the client window.	C-CSS-22000
14. Verify a message indicating the transfer server has received the message appears in the server window.	
15. Verify a message appears in the client window indicating that the client has obtained an UUID corresponding to the message object.	
16. Verify a message appears in the client window indicating that the client has obtained an UUID corresponding to the receiver.	C-CSS-22130
17. Verify a message appears in the client window indicating that the client has received acknowledgment that the message was delivered successfully to the intended receiver.	C-CSS-22110, C-CSS-22120
18. Verify a message appears in the client window indicating the return status of the message transmission.	C-CSS-22140
19. Verify a message appears in the client window indicating the message was retrieved from the queue.	C-CSS-22100
20. Kill the server process by issuing a kill signal in the server window. (e.g. **Ctrl-C)**	
21. Remove the server object from the cds namespace by highlighting the server object in the cdsbrowser and clicking on the **options->delete** entry button in the toolbar.	
22. In each window logout of dce by issuing the following commands: **kdestroy** **exit**	

4.4.1.2 Test Case 2: Synchronous Message Passing (TC010.002)

The purpose of this test is to verify that implementation one of the Message Passing Service has the capability to transmit messages/data via the synchronous message passage mode of communication.

Test Configuration:

Hardware: HP and Sun workstations/server HCI

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions

Data: N/A

Tools: sync_cli_hp, srv_hp, sync_cli_sun,srv_sun.

Test Input:

The name under which the test server application (e.g. srv_sun) will be registered in the cell directory service must be supplied to both the test server and client applications.

Test Output:

The client test driver should issue statements indicating that it is initializing the message service, creating a callback receiver and its associated queue, along with a callback object. It will then create a sending session list, add the callback receiver to the session list, and send the message. It will then wait until the message is processed and it has regained execution control. It should then display the uuid of the message object, the uuid of the receiver, and an acknowledgment that the message was delivered to the intended receiver. The server test driver should issue a statement indicating that the message was transferred.

Success Criteria:

The client application sends a message to the server and waits until the message transmission is complete. It then receives an acknowledgment indicating that the message was transmitted successfully.

Test Procedures:

Test Case ID: TC010.002	
Test Name: Synchronous Messaging Passing	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.
2. Create two sessions (i.e. open two xterm windows) and log into DCE in both by typing : **dce_login** *username password*	
3. In one window, start the cdsbrowser application in the background by typing: **cdsbrowser &**	Note the cdsbrowser application is available only on HP platforms. When executing this testcase on other platforms create a third session in step 2. In this window log into a HP machine in the same cell as the test machine, set your DISPLAY variable to point back to the test machine, and then execute step 3.
4. Start the test server application by typing the following in one of the test machine windows: **srv_hp** *<cds entry name>*	The cds entry should be placed in a directory in which the executor of the test has write permission. For example, in the relacell the following is a valid entry name: */.:/RelA/msg_server* where */.:/RelA/* is the directory and *msg_server* is the server name. A message stating "The server is now listening...." should appear in the server window indicating that the server is running.
5. Verify the server object has been registered in the CDS by clicking the **tree->update** button in the cdsbrowser's toolbar.	The server object should now be displayed inside the user-specified directory (e.g. underneath the */.:/RelA/* directory)
6. In the second window start the client application by typing the following: **cli_hp** *<cds entry name>*	The cds entry name should correspond to the cds entry name specified in step 4.

7. Verify a message indicating that the Message Passing service has been initialized appears in the client window.	
8. Verify a message indicating that a callback object has been created appears in the client window.	
9. Verify a message indicating that a callback receiver has been created appears in the client window.	
10. Verify a message indicating that a session list has been created appears in the client window.	
11. Verify a message indicating that the ordinary receiver has been added to the session list appears in the client window.	
12. Verify a message indicating that the message has been sent appears in the client window.	
13. Verify that the client test driver does not display any messages until step 14 is complete.	
14. Verify a message indicating the transfer server has received the message appears in the server window	
15. Verify a message appears in the client window indicating that the client has obtained an UUID corresponding to the message object.	
16. Verify a message appears in the client window indicating that the client has obtained an UUID corresponding to the receiver.	C-CSS-22130
17. Verify a message appears in the client window indicating that the client has received acknowledgment that the message was delivered successfully to the intended receiver.	C-CSS-22120
18. Verify a message appears in the client window indicating the return status of the message transmission.	
19. Kill the server process by issuing a kill signal in the server window. (e.g. **Ctrl-C)**	
20. Remove the server object from the cds namespace by highlighting the server object in the cdsbrowser and clicking on the **options->delete** entry button in the toolbar.	
21. In each window logout of dce by issuing the following commands: **kdestroy** **exit**	

4.4.1.3 Test Case 3: Transmission of Multiple Messages (TC010.003)

The purpose of this test is to verify that implementation one of the Message Passing Service is capable of transmitting multiple messages within an application. Specifically this test will demonstrate the queuing capability of the Message Passing Service.

Test Configuration:

Hardware: HP and Sun workstations/server

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions, ECS Common library functions.

Data: N/A

Tools: multi_cli_hp, srv_hp, multi_cli_sun, srv_sun.

Test Input:

A cds entry name must be supplied to the test client application. The user will be asked for the number of messages to send if it is not specified on the command line.

Test Output:

The client test driver should issue a statement indicating that the specified number of messages were sent and it is continuing its execution. The server test driver should issue statements indicating transferal of each message. For each message the client will then display its uuid, its receiver's uuid, an acknowledgment of its successful transmission to the intended receiver, and its return status.

Success Criteria:

The client application sends the messages to the server and continues executing. The client receives return information and an acknowledgment from the Message Service indicating each transmission was successful.

Test Procedures:

Test Case ID: TC010.003	
Test Name: Transmission of Multiple Messages	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.
2. Create two sessions (i.e. open two xterm windows) and log into DCE in both by typing : **dce_login** *username password*	
3. In one window, start the cdsbrowser application in the background by typing: **cdsbrowser &**	Note the cdsbrowser application is available only on HP platforms. When executing this testcase on other platforms create a third session in step 2. In this window log into a HP machine in the same cell as the test machine, set your DISPLAY variable to point back to the test machine, and then execute step 3.

4. Start the test server application by typing the following in one of the test machine windows: **srv_hp** *<cds entry name>*	The cds entry should be placed in a directory in which the executor of the test has write permission. For example, in the relacell the following is a valid entry name: */.:/RelA/msg_server* where */.:/RelA/* is the directory and *msg_server* is the server name. A message stating "The server is now listening...." should appear in the server window indicating that the server is running.
5. Verify the server object has been registered in the CDS by clicking the **tree->update** button in the cdsbrowser's toolbar.	The server object should now be displayed inside the user-specified directory (e.g. underneath the */.:/RelA/* directory)
6. In the second window start the client application by typing the following: **multi_cli_hp** *<cds entry name>*	The cds entry name should correspond to the cds entry name specified in step 4.
7. When prompted by the client test driver, enter the number of messages to transmit and press **Enter**.	
8. Verify the following message appears in the client window indicating the messages were issued: "*<#>* messages sent... continuing execution...."	C-CSS-22000, C-CSS-22090
9. Verify a message appears in the server window for each message sent, indicating the message was received by the transfer server.	
10. Verify a message appears in the client window indicating that the client has obtained an UUID corresponding to the message object for each message sent.	
11. Verify a message appears in the client window indicating that the client has obtained an UUID corresponding to the receiver for each message sent.	C-CSS-22130
12. Verify a message appears in the client window indicating that the client has received acknowledgment that each message was delivered successfully to the intended receiver.	C-CSS-22120, C-CSS-22110
13. Verify a message appears in the client window indicating the return status of each message transmission.	C-CSS-22140
14. Verify a message appears in the client window indicating the messages were retrieved from the queue.	C-CSS-22100
15. Kill the server process by issuing a kill signal in the server window. (e.g. **Ctrl-C)**	
16. Remove the server object from the cds namespace by highlighting the server object in the cdsbrowser and clicking on the **options->delete** entry button in the toolbar.	
17. In each window logout of dce by issuing the following commands: **kdestroy** **exit**	

4.4.1.4 Test Case 4: Transmission of Messages to Multiple Receivers (TC010.004)

The purpose of this test is to verify that implementation one of the Message Passing Service is capable of transmitting messages to multiple receivers . Specifically this test will demonstrate the Message Passing Service's ability to designate multiple receivers for asynchronous messages and its support of multiple message queues.

Test Configuration:

Hardware: HP and Sun workstations/server HCI, Inter networking subsystem CI (LAN Network resources)-INCI

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions, ECS Common library functions.

Data: N/A

Tools: mrec_cli_hp, srv_hp, mrec_cli_sun, srv_sun.

Test Input:

A cds entry name must be supplied to the test client application. The user will be asked for the number of messages to send if it is not specified on the command line along with the number of receivers to create.

The client test driver should issue statements indicating that it is initializing the message service, creating a callback receiver and its associated queue, along with a callback object. It will then create a sending session list, add the callback receiver to the session list, send the message and it continue its execution. It should then display the uuid of the message object, the uuid of the receiver, an acknowledgment that the message was delivered to the intended receiver, and the return status of the message transmission. The server test driver should issue a statement indicating that the message was transferred.

Test Output:

The client test driver should issue statements indicating that it is initializing the message service, creating the specified number of callback receivers and their associated queues, along with a callback object. It will then create a sending session list for each receiver and an additional list that will contain all the receivers. The callback receivers will then be attached to the appropriate session lists. Statements indicating that the specified number of messages were sent to the specified number of receivers will be issued and the client will continue its execution. The server test driver should issue statements indicating transferal of each message. For each message the client will then display its uuid, its receiver's uuid, an acknowledgment of its successful transmission to the intended receiver, and its return status.

Success Criteria:

The client application sends the messages to the receivers and continues executing. The client receives return information and an acknowledgment from the Message Service indicating each transmission was successful and successfully retrieves the messages from the appropriate queues.

Test Procedures:

Test Case ID: TC010.004	
Test Name: Transmission of Messages to Multiple Receivers	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.
2. Create two sessions (i.e. open two xterm windows) and log into DCE in both by typing : **dce_login** *username password*	
3. In one window, start the cdsbrowser application in the background by typing: **cdsbrowser &**	Note the cdsbrowser application is available only on HP platforms. When executing this testcase on other platforms create a third session in step 2. In this window log into a HP machine in the same cell as the test machine, set your DISPLAY variable to point back to the test machine, and then execute step 3.
4. Start the test server application by typing the following in one of the test machine windows: **srv_hp** *<cds entry name>*	The cds entry should be placed in a directory in which the executor of the test has write permission. For example, in the relacell the following is a valid entry name: */.:/RelA/msg_server* where */.:/RelA/* is the directory and *msg_server* is the server name. A message stating "The server is now listening...." should appear in the server window indicating that the server is running.
5. Verify the server object has been registered in the CDS by clicking the **tree->update** button in the cdsbrowser's toolbar.	The server object should now be displayed inside the user-specified directory (e.g. underneath the */.:/RelA/* directory)
6. In the second window start the client application by typing the following: **mrec_cli_hp** *<cds entry name>*	The cds entry name should correspond to the cds entry name specified in step 4.
7. When prompted by the client test driver, enter the number of messages to transmit and press **Enter**.	
8. When prompted by the client test driver, enter the number of receivers to send the message to and press **Enter**.	
9. Verify a message indicating that the Message Passing service has been initialized appears in the client window.	
10. Verify a message indicating that a callback object has been created appears in the client window.	

11. Verify a message indicating that the specified number of callback receivers have been created appears in the client window.	Each receiver has its own incoming queue allowing different groups of processes to use different queues. C-CSS-22050
12. Verify a message indicating that appropriate number of session lists have been created appears in the client window.	C-CSS-22040
13. Verify a message indicating that the callback receivers have been added to the appropriate session list appears in the client window.	
14. Verify the following message appears in the client window indicating the messages were issued: "<#> messages sent... continuing execution...."	C-CSS-22000, C-CSS-22090
15. Verify a message appears in the server window for each message sent, indicating the message was received by the transfer server.	
Verify a message appears in the client window indicating that the client has obtained an UUID corresponding to the message object for each message sent.	
Verify a message appears in the client window indicating that the client has obtained an UUID corresponding to the receiver for each message sent.	C-CSS-22130
Verify a message appears in the client window indicating that the client has received acknowledgment that each message was delivered successfully to the intended receiver.	C-CSS-22110, C-CSS-22120
Verify a message appears in the client window indicating the return status of each message transmission.	C-CSS-22140
Verify a message appears in the client window indicating the messages were retrieved from the appropriate queues.	C-CSS-22100
16. Kill the server process by issuing a kill signal in the server window. (e.g. **Ctrl-C)**	
17. Remove the server object from the cds namespace by highlighting the server object in the cdsbrowser and clicking on the **options->delete** entry button in the toolbar.	
18. In each window logout of dce by issuing the following commands: **kdestroy** **exit**	

4.4.1.5 Test Case 5: Retransmission of Unsuccessful Message (TC010.005)

The purpose of this test is to verify that implementation one of the Message Passing Service can defer sending a message to a receiver when the receiver is not active. Specifically, this test will

demonstrate the Message Passing Service's ability to store undeliverable messages and retry sending it a specified number of times at specified time intervals.

Test Configuration:

Hardware: HP and Sun workstations/server

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions, ECS Common library functions.

Data: N/A

Tools: ret_cli_hp, srv_hp, ret_cli_sun, srv_sun.

Test Input:

The name under which the test server application (e.g. srv_sun) will be registered in the cell directory service must be supplied to both the test server and client applications. The client application will also prompt the user for the number of times to try sending the message as well as the time interval between transmission attempts.

Test Output:

The client test driver should issue statements indicating that a message was sent and it is continuing its execution. Subsequently, messages indicating how many transmissions have been attempted and the system time of the current attempt are displayed. When the message transmission is completed the uuid of the message object, the uuid of the receiver and the return status of the message transmission will be displayed. The server test driver should issue a statement indicating that the message was transferred.

Success Criteria:

The client application sends a message to the server and continues executing. The Message Service retransmits the message until it is successful in connecting to the server. Upon completion of the message transmission the client receives return information and an acknowledgment from the message passing service indicating that the message thread executed successfully. It then retrieves the message from the queue.

Test Procedures:

Test Case ID: TC010.005	
Test Name: Retransmission of Unsuccessful Message	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.
2. Create two sessions (i.e. open two xterm windows) and log into DCE in both by typing : **dce_login** *username password*	
3. In one window, start the cdsbrowser application in the background by typing: **cdsbrowser &**	Note the cdsbrowser application is available only on HP platforms. When executing this testcase on other platforms create a third session in step 2. In this window log into a HP machine in the same cell as the test machine, set your DISPLAY variable to point back to the test machine, and then execute step 3.
4. Start the test client application by typing the following in one of the test machine windows: **ret_cli_hp** *<cds entry name>*	The cds entry name should correspond to the cds entry name that will be used in registering the server application. See step11 for more details.
5. When prompted by the client test driver, enter the number of times to try sending the message and press **Enter**.	Enter a number large enough so that there will be time to start the server application in the second window before the Message Service stops trying to retransmit the message.
6. When prompted by the client test driver, enter the time interval to wait between transmission attempts and press **Enter**.	The time interval should be in accordance with the number of retransmission attempts so that the server application can be started before the Message Service stops trying to retransmit the message.
7. Verify the following message appears in the client window indicating the message was issued: "Sending message to *receiver...*"	C-CSS-22090
8. Verify the following message appears in the client window indicating the client has regained control of the process's execution: "Continuing execution..."	C-CSS-22000
9. Verify the following message appears in the client window indicating the message thread is in its initial attempt at execution: "Current transmission attempt: 1." "Time of attempt:*<current time>*"	
10. Verify the following message appears in the client window indicating the message thread could not locate the server object: "**DCE Exception: entry not found (dce/rpc)"	This DCE Exception occurs because the server application has not been registered in the cell's namespace. A message of this type will occur every time a transmission attempt fails because the object could not be found.

11. In the second window start the test server application by typing the following: **srv_hp** *<cds entry name>*	The cds entry should be placed in a directory in which the executor of the test has write permission. For example, in the relacell the following is a valid entry name: */.:/RelA/msg_server* where */.:/RelA/* is the directory and *msg_server* is the server name. A message stating "The server is now listening...." should appear in the server window indicating that the server is running.
12. Verify the server object has been registered in the CDS by clicking the **tree->update** button in the cdsbrowser's toolbar.	The server object should now be displayed inside the user-specified directory (e.g. underneath the */.:/RelA/* directory)
13. Verify a message indicating the transfer server has received the message appears in the server window.	
Verify a message appears in the client window indicating that the client has obtained an UUID corresponding to the message object.	
Verify a message appears in the client window indicating that the client has obtained an UUID corresponding to the receiver.	C-CSS-22130
Verify a message appears in the client window indicating that the client has received acknowledgment that the message was delivered successfully to the intended receiver.	C-CSS-22110, C-CSS-22120
Verify a message appears in the client window indicating the return status of the message transmission.	C-CSS-22140
Verify a message appears in the client window indicating the message was retrieved from the queue.	C-CSS-22100
14. Kill the server process by issuing a kill signal in the server window. (e.g. **Ctrl-C)**	
15. Remove the server object from the cds namespace by highlighting the server object in the cdsbrowser and clicking on the **options->delete** entry button in the toolbar.	
16. In each window logout of dce by issuing the following commands: **kdestroy** **exit**	

4.4.1.6 Test Case 6: Message Passing to Non-Existent Receiver (TC010.006)

The purpose of this test is to verify that implementation one of the Message Passing Service takes appropriate action when it is unsuccessful in transmitting a message in its allotted time interval. Specifically this test will demonstrate the Message Passing Service's ability purge undeliverable messages from the message queue and to log failed attempt event messages to the MSS management agents The failed attempt status information will be stored for the client to retrieve at its convenience.

Test Configuration:

Hardware: HP and Sun workstations/server HCI, Inter networking subsystem CI (LAN Network resources)-INCI

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions, ECS Common library functions.

Data: N/A

Tools: ret_cli_hp,ret_cli_sun.

Test Input:

A cds entry name must be supplied to the test client application. The client application will also prompt the user for the number of times to try sending the message as well as the time interval between transmission attempts.

Test Output:

The client test driver should issue statements indicating that a message was sent and it is continuing its execution. Subsequently, messages indicating how many transmissions have been attempted and the system time of the current attempt are displayed. When the message object exhausts its allotted transmission attempts an error message will appear in the client window. This message corresponds to the entry that will be logged to the MSS management agent. The uuid of the message object and the intended receiver will be displayed along with the return status of the message transmission.

Success Criteria:

The client application sends a message to the server and continues executing. The Message Service will retransmit the message until it exhausts its allotted number of transmission attempts. The Message Service will log the transmission failure to the MSS management agent and the corresponding log message will appear in the client window. The client retrieves the return information indicating that the execution of the message thread was unsuccessful.

Test Procedures:

Test Case ID: TC010.006	
Test Name: Message Passing to Non-Existent Receiver	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.

2. Create a session (i.e. a xterm windows) and log into DCE by typing : **dce_login** *username password*	
3. In the window start the client application by typing the following: **ret_cli_hp** *<cds entry name>*	Normally the cds entry name would correspond to the cds entry name under which the server application was registered. Since the server does not exist in this case an entry name should be used that is not registered in the cell's name space.
4. When prompted by the client test driver, enter the number of times to try sending the message and press **Enter**.	
5. When prompted by the client test driver, enter the time interval to wait between transmission attempts and press **Enter**.	
6. Verify the following message appears in the client window indicating the message was issued: "Sending message to *receiver...*"	C-CSS-22070
7. Verify the following message appears in the client window indicating the client has regained control of the process's execution: "Continuing execution..."	C-CSS-22000
8. Verify the following message appears in the client window indicating the message thread is in its initial attempt at execution: "Current transmission attempt: 1." "Time of attempt:*<current time>*"	Messages of this format will continue to appear in the client window until the Message Service has attempted to transmit the message the number of times specified in step 4. C-CSS-22150
9. Verify the following message appears in the client window indicating the message thread could not locate the server object: "**DCE Exception: entry not found (dce/rpc)"	This DCE Exception occurs because the server application has not been registered in the cell's namespace. A message of this type will occur every time a transmission attempt fails because the object could not be found.
Verify a message appears in the client window indicating the failed execution of the message transmission.	The text of this message corresponds to the text that will be logged to the MSS Management agent. C-CSS-22065
Verify a message appears in the client window indicating that the client has obtained an UUID corresponding to the message object.	
Verify a message appears in the client window indicating that the client has obtained an UUID corresponding to the receiver.	
Verify a message appears in the client window indicating the return status of the message transmission.	
Ensure the message has been removed from the outgoing queue by verifying it is not listed in the outgoing queue persistence file.	The name of the file will be displayed in the test driver. This file contains a list of all the messages currently in the outgoing queue. C-CSS-22060
10. In the window logout of dce by issuing the following commands: **kdestroy** **exit**	

4.4.1.7 Test Case 7: Persistence and Recovery (TC010.007)

The purpose of this test is to verify that implementation one of the Message Passing Service is capable of restoring its incoming and outgoing queues in the event of an application or system failure. Specifically this test will demonstrate the persistence capabilities of the Message Passing Service and its ability to use this persistence to implement a recovery mechanism in case of failure.

Test Configuration:

Hardware: HP and Sun workstations/server

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions, ECS Common library functions.

Data: N/A

Tools: fail_cli_hp, rec_cli_hp, srv_hp, fail_cli_sun, rec_cli_sun, srv_sun.

Test Input:

A cds entry name must be supplied to the test client applications. The user will be asked for the number of messages to send if it is not specified on the command line along with the number of receivers to create.

Test Output:

The client test driver should issue statements indicating that it is initializing the message service, creating the specified number of callback receivers and their associated queues, along with a callback object. It will then create a sending session list for each receiver and an additional list that will contain all the receivers. The callback receivers will then be attached to the appropriate session lists. Statements indicating that the specified number of messages were sent to the specified number of receivers will be issued and the client will continue its execution. The client application will then exit. When the subsequent client application is run it will recreate the receivers and attempt to recover the queues created in the earlier driver. The server test driver should issue statements indicating transferal of each message. For each message the client will then display its uuid, its receiver's uuid, an acknowledgment of its successful transmission to the intended receiver, and its return status.

Success Criteria:

The first client application sends the messages to the receivers and exits. The second client application restores the incoming and outgoing queues and continues execution. It then receives return information and an acknowledgment from the Message Service indicating each transmission was successful and successfully retrieves the messages from the appropriate queues.

Test Procedures:

Test Case ID: TC010.007	
Test Name: Persistence and Recovery	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.
2. Create two sessions (i.e. open two xterm windows) and log into DCE in both by typing : **dce_login** *username password*	
3. In one window, start the cdsbrowser application in the background by typing: **cdsbrowser &**	Note the cdsbrowser application is available only on HP platforms. When executing this testcase on other platforms create a third session in step 2. In this window log into a HP machine in the same cell as the test machine, set your DISPLAY variable to point back to the test machine, and then execute step 3.
4. In the second window start the test server application by typing the following: **srv_hp** *<cds entry name>*	The cds entry should be placed in a directory in which the executor of the test has write permission. For example, in the relacell the following is a valid entry name: */.:/RelA/msg_server* where */.:/RelA/* is the directory and *msg_server* is the server name. A message stating "The server is now listening...." should appear in the server window indicating that the server is running.
5. In the second window start the client application by typing the following: **fail_cli_hp** *<cds entry name>*	
6. When prompted by the client test driver, enter the number of messages to transmit and press **Enter**.	
7. When prompted by the client test driver, enter the number of receivers to send the message to and press **Enter**.	
8. Verify a message indicating that the Message Passing service has been initialized appears in the client window.	
9. Verify a message indicating that a callback object has been created appears in the client window.	
10. Verify a message indicating that the specified number of callback receivers have been created appears in the client window.	Each receiver has its own incoming queue allowing different groups of processes to use different queues. C-CSS-22050
11. Verify a message indicating that appropriate number of session lists have been created appears in the client window.	
12. Verify a message indicating that the callback receivers have been added to the appropriate session list appears in the client window.	

13. Verify the following message appears in the client window indicating the messages were issued: "*<#>* messages sent... continuing execution...."	Half of the messages will be sent at once and the other half will be sent 15 seconds later. This will allow half the messages to be transferred to their appropriate receiver's incoming queue and the other half will be in the outgoing queue when the test exits.
14. Verify a message indicating the client application is exiting appears in the client window.	
15. Verify a message appears in the server window for half of the message sent, indicating the message was received by the transfer server.	
16. Kill the server process by issuing a kill signal in the server window. (e.g. **Ctrl-C)**	
17. Remove the server object from the cds namespace by highlighting the server object in the cdsbrowser and clicking on the **options->delete** entry button in the toolbar.	
18. Start the test server application by typing the following in one of the test machine windows: **srv_hp** *<cds entry name>*	The cds entry should be placed in a directory in which the executor of the test has write permission. For example, in the relacell the following is a valid entry name: */.:/RelA/msg_server* where */.:/RelA/* is the directory and *msg_server* is the server name. A message stating "The server is now listening...." should appear in the server window indicating that the server is running.
19. In the second window start the client application by typing the following: **rec_cli_hp** *<cds entry name>*	The cds entry name should correspond to the cds entry name that was used in step 19.
20. Verify a message indicating the callback receivers are being recreated appears in the client window.	This is not part of the Message Service functionality.
21. Verify a message indicating the queues have been recovered appears in the client window.	
22. Verify a message appears in the client window indicating that the client has obtained an UUID corresponding to the message object for each message sent.	
23. Verify a message appears in the client window indicating that the client has obtained an UUID corresponding to the receiver for each message sent.	
24. Verify a message appears in the client window indicating that the client has received acknowledgment that each message was delivered successfully to the intended receiver.	
25. Verify a message appears in the client window indicating the return status of each message transmission.	
26. Verify a message appears in the client window indicating the messages were retrieved from the appropriate queues.	

27. Kill the server process by issuing a kill signal in the server window. (e.g. **Ctrl-C)**	
28. Remove the server object from the cds namespace by highlighting the server object in the cdsbrowser and clicking on the **options->delete** entry button in the toolbar.	
29. In each window logout of dce by issuing the following commands: **kdestroy** **exit**	

4.4.2 Directory Naming Service Thread (TC011)

The purpose of this thread is to verify the functionality of the Directory/Naming Service, which is used to uniquely associate a name with resources/principals along with their attributes. This thread will demonstrate the naming service functionality which consists of the *Context* and *Element (Attribute/Value(s))* classes which provides the overall naming functionality to store/retrieve information into/from the namespace via name service specifications; X/Open's *X.500* and the internet's *DNS*. Included in this thread is to demonstrate the capability of *extensibility*. CSS Naming Service uses both XDS/XOM internal interfaces for the enterprise GDS/CDS namespaces. These interfaces open communication throughout the network between namespaces.

4.4.2.1 Test Case 1: Create Entry Names (TC011.001)

This test case demonstrates that the Directory naming Service permits application users to Create Entry Names via using the AddCtx method to add each context key or by loading a full context constructor string.

Requirement C-CSS-20025(b) is satisfied through Inspection. Updating the namespace *manually* is a DCE COTS feature, reference the OSF DCE Administration Guide-Core Components chp. 6 for Using the CDS Control Program and see the OSF DCE Administration Reference for complete cdscp command descriptions.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE/OODCE, Cell Directory Service, cdsbrowser, Clearcase, DNS

Data: Entry names

Test Input:

Test driver main_hp or main_sun used to create entry names using both AddCtx method and constructor to load full context string.

Test Output:

The Directory Naming Service displays all entry names, including entries created.

Success Criteria:

Entry names created in the namespace successfully display their names.

Test Procedures:

Test Case ID: TC011.001	
Test Name: Create Entry Names	
Test Steps:	Comments
1. Login to **client.**	
2. Perform DCE login. **"dce_login <username>"**	
3. Enter DCE **password**	
4. Enter: **cd /usr/testa/IT/tools/CSS/DN/<platform>**	substitute sun for hp, if testing sun platform
5. Enter: **cdsbrowser &**	Used to verify directory naming activity
6. Enter: **main_<**platform>	platform is hp or sun
7. Monitor msgs for **Name /.:/RelA/Entry2 Loaded using EcDnCompositeName AddCtx method.**	Press: Return until msg is displayed. AddCtx method loads each context key separately. C-CSS-20000 (b)
8. Press: **Tree** Press: **Update**	Refresh screen from cdsbrowser
9. Verify entry name added: **/.:/RelA/Entry2**	Verify Entry Name Created using cdsbrowser. C-CSS-20000 (a)
10. Monitor msgs for **Name /.:/RelA/test2 Loaded using EcDnCompositeName CONSTRUCTOR (a string.**	Press: Return until msg is displayed. CONSTRUCTOR method C-CSS-20000 (b)
11. Press: **Tree** Press: **Update**	Refresh screen from cdsbrowser
12. Verify entry name added: **/.:/RelA/test2**	Verify Entry Name Created using cdsbrowser. C-CSS-20000 (a), C-CSS-20060, C-CSS-20110
13. Press: ctrl-c To exit from driver	
14. From cdsbrowser: Highlight new entry **/.:/RelA/Entry2**	
15. From cdsbrowser: Press: Action Press: Delete Entry	Delete new entry using cdsbrowser

4.4.2.2 Test Case 2: Add/Get Element to Entry Name (TC011.002)

This test case will demonstrate that the Directory/Naming Service can Add and Get Elements to and from an object entry.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE/OODCE, Cell Directory Service, cdsbrowser, Clearcase, DNS

Data: Entry names with attributes and values.

Test Input:

Test driver main_hp or main_sun used to Add and Get elements.

Test Outputs

The Directory/Naming Service should display all entry names including any element added to the entry.

Success Criteria

Element is added to the entry name and successfully displayed.

Test Procedures:

Test Case ID: TC011.002	
Test Name: Add/Get Element to Entry Name	
Test Steps:	Comments
1. Login to **client.**	
2. Perform DCE login. **"dce_login <username>"**	
3. Enter DCE **password**	
4. Enter: **cd /usr/testa/IT/tools/CSS/DN/<platform>**	substitute sun for hp, if testing sun platform
5. Enter: **cdsbrowser &**	Used to verify directory naming activity
6. Enter: **main_<**platform>	platform is hp or sun
7. Monitor msgs for **Adding Element will set the Attribute...**	Press: Return until msg is displayed. C-CSS-20000 (c)
8. From cdsbrowser: Press: **Tree** Press: **Update**	Refresh screen from cdsbrowser
9. Verify entry name added: **/.:/RelA/Entry2**	Verify Entry Name Added using cdsbrowser C-CSS-20070
10. From cdsbrowser: **Double-click /.:/RelA/Entry2**	To access Attribute lists
11. From cdsbrowser: **Select attribute added** Press: **Display**	Gets and displays Attribute(s)
12. Verify **Attribute** CSSAttr added	List Attribute C-CSS-20000 (e)
13. From cdsbrowser: **Double-click CSSAttr** to list values	Gets and displays an Attributes values
14. Verify added **Values** CSSValue0, 1, 2 are displayed	List Values C-CSS-20000 (f)
15. Press: ctrl-c To exit from driver	
16. From cdsbrowser: Highlight **/.:/RelA/Entry2** entry added	
17. From cdsbrowser: Press: Action Press: Delete Entry	Delete new entry using cdsbrowser

4.4.2.3 Test Case 3: ADD Additional Values to Element (TC011.003)

This test case will demonstrate that additional object element values can be added to an existing list of element values.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE/OODCE, Cell Directory Service, cdsbrowser, Clearcase, DNS

Data: Entry names with attributes and values.

Test Input:

Test driver main_hp or main_sun used to Add additional values to the element.

Test Outputs

The Directory/Naming Service must demonstrate that additional values can be added to an existing list of values successfully.

Success Criteria

The Directory/Naming Service additional element values are added to the existing element values successfully.

Test Procedures:

Test Case ID: TC011.003	
Test Name: ADD Additional Values to Element	
Test Steps:	Comments
1. Login to **client.**	
2. Perform DCE login. **"dce_login <username>"**	
3. Enter DCE **password**	
4. Enter: **cd /usr/testa/IT/tools/CSS/DN/<platform>**	substitute sun for hp, if testing sun platform
5. Enter: **cdsbrowser &**	Used to verify directory naming activity
6. Enter: **main_<**platform>	platform is hp or sun
7. Monitor msgs for **Add more values to the element ...**	Press: Return until msg is displayed. C-CSS-20000 (f)
8. From cdsbrowser: Press: **Tree** Press: **Update**	Refresh screen from cdsbrowser
9. Verify entry name added: **/.:/RelA/Entry2**	Verify Entry Name Added using cdsbrowser
10. From cdsbrowser: **Double-click /.:/RelA/Entry2**	To access Attribute lists
11. From cdsbrowser: **Select attribute** Press: **Display**	Gets and displays Attribute CSSAttr
12. From cdsbrowser: **Double-click CSSAttr** to list values	Gets and displays additional values added
13. Verify additional **Values** CSSValue3, 4, 5 are displayed	List additional Values C-CSS-20000 (f)
14. Press: ctrl-c To exit from driver	
15. From cdsbrowser: Highlight **/.:/RelA/Entry2** entry added	
16. From cdsbrowser: Press: Action Press: Delete Entry	Delete new entry using cdsbrowser

4.4.2.4 Test Case 4: Delete a Value from an Element (TC011.004)

This test case must demonstrate that the Directory/Naming Service can delete a value from an object entry element.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE/OODCE, Cell Directory Service, cdsbrowser, Clearcase, DNS

Data: Entry names with attributes and values.

Test Input:

Test driver main_hp or main_sun used to delete a value from an object entry element.

Test Outputs

The specific object entry element value is completely deleted.

Success Criteria

The Directory/Naming Service must demonstrate that a value contained in an element list can be successfully deleted, from the directory.

Test Procedures:

Test Case ID: TC011.004	
Test Name: Delete a Value from an Element	
Test Steps:	Comments
1. Login to **client.**	
2. Perform DCE login. **"dce_login <username>"**	
3. Enter DCE **password**	
4. Enter: **cd /usr/testa/IT/tools/CSS/DN/<platform>**	substitute sun for hp, if testing sun platform
5. Enter: **cdsbrowser &**	Used to verify directory naming activity
6. Enter: **main_<**platform>	platform is hp or sun
7. Monitor msgs for **Get the List of Values for the Attribute of the Element Read ...**	Press: Return until msg is displayed.
8. From cdsbrowser: Press: **Tree** Press: **Update**	Refresh screen from cdsbrowser
9. From cdsbrowser: **Double-click /.:/RelA/Entry2**	To access Attribute lists
10. From cdsbrowser: **Select attribute** Press: **Display**	Gets and displays Attribute CSSAttr
11. From cdsbrowser: **Double-click CSSAttr** to list values	Gets and displays additional values added
12. Verify additional **Values** CSSValue3, 4, 5 are displayed	List additional Values
13. Monitor msgs for **Delete a Value from an element ...**	Press: Return until msg is displayed.
14. Verify that only the additional value **CSSValue5** is deleted.	C-CSS-20000 (d)
15. Press: ctrl-c To exit from driver	
16. From cdsbrowser: Highlight **/.:/RelA/Entry2** entry added	
17. From cdsbrowser: Press: Action Press: Delete Entry	Delete new entry using cdsbrowser

4.4.2.5 Test Case 5: Partial Delete of a List of Values (TC011.005)

This test case must demonstrate that the Directory/Naming Service can delete a part of a list of values from an object entry element.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE/OODCE, Cell Directory Service, cdsbrowser, Clearcase, DNS

Data: Entry names with attributes and values.

Test Input

Test driver main_hp or main_sun used to delete a list of values from an object entry element.

Test Outputs

The object entry element value list is completely deleted.

Success Criteria

The Directory/Naming Service must demonstrate that a list of values contained in an element list can be successfully deleted, from the directory.

Test Procedures:

Test Case ID: TC011.005	
Test Name: Partial Delete of a List of Values	
Test Steps:	Comments
1. Login to **client.**	
2. Perform DCE login. **"dce_login <username>"**	
3. Enter DCE **password**	
4. Enter: **cd /usr/testa/IT/tools/CSS/DN/<platform>**	substitute sun for hp, if testing sun platform
5. Enter: **cdsbrowser &**	Used to verify directory naming activity
6. Enter: **main_<**platform>	platform is hp or sun
7. Monitor msgs for **Get the List of Values for the Attribute of the Element Read ...**	Press: Return until msg is displayed.
8. From cdsbrowser: Press: **Tree** Press: **Update**	Refresh screen from cdsbrowser
9. From cdsbrowser: **Double-click /.:/RelA/Entry2**	To access Attribute lists
10. From cdsbrowser: **Select attribute** Press: **Display**	Gets and displays Attribute CSSAttr
11. From cdsbrowser: **Double-click CSSAttr** to list values	Gets and displays additional values added
12. Verify additional **Values** CSSValue3, 4, 5 are displayed	List additional Values
13. Monitor msgs for **Delete a List of Values from an Element ...**	Press: Return until msg is displayed.
14. Verify that the value list **CSSValue3, 4, 5** is deleted.	C-CSS-20000 (d)
15. Press: ctrl-c To exit from driver	

16. From cdsbrowser: Highlight **/.:/RelA/Entry2** entry added	
17. From cdsbrowser: Press: Action Press: Delete Entry	Delete new entry using cdsbrowser

4.4.2.6 Test Case 6: User Defined Attribute Types (TC011.006)

This test case will demonstrate the capability of the user to defined a minimum of 20 attribute types for application users to store/retrieve attribute information.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE/OODCE, Cell Directory Service, cdsbrowser, Clearcase, DNS

Data: Entry names with attributes and values.

Test Input:

Test driver main_hp or main_sun used define mutiple atttribute types.

Test Outputs

The Directory/Naming Service must demonstrate that the user to can define a minimum of 20 attribute types for application users to store/retrieve attribute information.

Success Criteria

The Directory/Naming Service successfully allows the user to defined a minimum of 20 attribute types for application users to store/retrieve attribute information.

Test Procedures:

Test Case ID: TC011.006	
Test Name: User Defined Attribute Types	
Test Steps:	Comments
1. Login to **client.**	
2. Perform DCE login. **"dce_login <username>"**	
3. Enter DCE **password**	
4. Enter: **cd /usr/testa/IT/tools/CSS/DN/<platform>**	substitute sun for hp, if testing sun platform
5. Enter: **cdsbrowser &**	Used to verify directory naming activity
6. Enter: **main_<**platform>	platform is hp or sun
7. Monitor msgs for **Add another Attribute to element ...**	Press: Return until msg is displayed.
8. From cdsbrowser: Press: **Tree** Press: **Update**	Refresh screen from cdsbrowser
9. Verify entry name added: **/.:/RelA/Entry2**	Verify Entry Name Added using cdsbrowser
10. From cdsbrowser: **Double-click /.:/RelA/Entry2**	To access Attribute lists
11. From cdsbrowser: **Select attribute** Press: **Display**	Gets and displays Attribute MSSAttr
12. Monitor msgs for **Read this Attribute ...**	Press: Return until msg is displayed.
13. From cdsbrowser: **Double-click MSSAttr** to list values	Gets and displays types

14. Verify a list of 20 attribute types are defined MSSValue0, 1,..19	List of types C-CSS-20090
15. Press: ctrl-c To exit from driver	
16. From cdsbrowser: Highlight **/.:/RelA/Entry2** entry added	
17. From cdsbrowser: Press: Action Press: Delete Entry	Delete new entry using cdsbrowser

4.4.2.7 Test Case 7: Modify a Value (TC011.007)

This test case will demonstrate that the Directory/Naming Service can facilitate modifying attribute values from an object entry element.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE/OODCE, Cell Directory Service, cdsbrowser, Clearcase, DNS

Data: Entry names with attributes and values.

Test Input:

Test driver main_hp or main_sun used to modify a value from an object element.

Test Outputs

The element value is modified from one value to another value.

Success Criteria

The Directory/Naming Service must demonstrate that an element value can be successfully modified from one value to another.

Test Procedures:

Test Case ID: TC011.007	
Test Name: Modify a Value	
Test Steps:	Comments
1. Login to **client.**	
2. Perform DCE login. **"dce_login <username>"**	
3. Enter DCE **password**	
4. Enter: **cd /usr/testa/IT/tools/CSS/DN/<platform>**	substitute sun for hp, if testing sun platform
5. Enter: **cdsbrowser &**	Used to verify directory naming activity
6. Enter: **main_<**platform>	platform is hp or sun
7. From cdsbrowser: Press: **Tree** Press: **Update**	Refresh screen from cdsbrowser
8. From cdsbrowser: **Double-click /.:/RelA/Entry2**	To access Attribute lists
9. From cdsbrowser: **Select attribute** Press: **Display**	Gets and displays Attribute MSSAttr
10. From cdsbrowser: **Double-click MSSAttr** to list values	Gets and displays values list
11. Verify Values MSSValue0, 1, ... , 19 4 are displayed	List of Values
12. Monitor msgs for **Modify a Value of Attribute ...**	Press: Return until msg is displayed.

13. Verify that the value list **MSSValue3 is modified to ISSValue0**.	Value Modified. C-CSS-20025(a), C-CSS-20020
14. Press: ctrl-c To exit from driver	
15. From cdsbrowser: Highlight **/.:/RelA/Entry2** entry added	
16. From cdsbrowser: Press: Action Press: Delete Entry	Delete new entry using cdsbrowser

4.4.2.8 Test Case 8: Modify an Attribute (TC011.008)

This test case will demonstrate that the Directory/Naming Service can facilitate modifying attribute values from an object entry element.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE/OODCE, Cell Directory Service, cdsbrowser, Clearcase, DNS

Data: Entry names with attributes and values.

Test Input:

Test driver main_hp or main_sun used to modify the attribute of an object element.

Test Outputs

The element attribute is modified from one attribute name to another.

Success Criteria

The Directory/Naming Service must demonstrate that an element value can be successfully modified from one value to another.

Test Procedures:

Test Case ID: TC011.008	
Test Name: Modify an Attribute	
Test Steps:	Comments
1. Login to **client.**	
2. Perform DCE login. **"dce_login <username>"**	
3. Enter DCE **password**	
4. Enter: **cd /usr/testa/IT/tools/CSS/DN/<platform>**	substitute sun for hp, if testing sun platform
5. Enter: **cdsbrowser &**	Used to verify directory naming activity
6. Enter: **main_<**platform>	platform is hp or sun
7. From cdsbrowser: Press: **Tree** Press: **Update**	Refresh screen from cdsbrowser
8. From cdsbrowser: **Double-click /.:/RelA/Entry2**	To access Attribute lists
9. From cdsbrowser: **Select attribute** Press: **Display**	Gets and displays Attribute MSSAttr
10. Verify Attribute **MSSAttr** is displayed	List of Values
11. Monitor msgs for **Modify an Attribute ...**	Press: Return until msg is displayed.
12. Verify that the value list **MSSAttr is modified to NewAttr**.	Attribute Modified. C-CSS-20020, C-CSS-20025(a)

13. Press: ctrl-c To exit from driver	
14. From cdsbrowser: Highlight **/.:/RelA/Entry2** entry added	
15. From cdsbrowser: Press: Action Press: Delete Entry	Delete new entry using cdsbrowser

4.4.2.9 Test Case 9: Delete an Attribute from an Entry (TC011.009)

This test case must demonstrate that the Directory/Naming Service can delete an attribute from an object entry element.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE/OODCE, Cell Directory Service, cdsbrowser, Clearcase, DNS

Data: Entry names with attributes and values.

Test Input:

Test driver main_hp or main_sun used to delete an attribute from an object entry element.

Test Outputs

The object entry attribute is completely deleted.

Success Criteria

The Directory/Naming Service must demonstrate that an attribute can be successfully deleted, from the directory.

Test Procedures:

Test Case ID: TC011.009	
Test Name: Delete an Attribute from an Entry	
Test Steps:	Comments
1. Login to **client.**	
2. Perform DCE login. **"dce_login <username>"**	
3. Enter DCE **password**	
4. Enter: **cd /usr/testa/IT/tools/CSS/DN/<platform>**	substitute sun for hp, if testing sun platform
5. Enter: **cdsbrowser &**	Used to verify directory naming activity
6. Enter: **main_<**platform>	platform is hp or sun
7. Monitor msgs for **Read Attribute Again and Verify Changes .**	Press: Return until msg is displayed.
8. From cdsbrowser: Press: **Tree** Press: **Update**	Refresh screen from cdsbrowser
9. From cdsbrowser: **Double-click /.:/RelA/Entry2**	To access Attribute lists
10. From cdsbrowser: **Select attribute** Press: **Display**	Gets and displays Attribute MSSAttr
11. Monitor msgs for **Delete an Atttribute ...**	Press: Return until msg is displayed.
12. Verify attribute **MSSAttr** is deleted.	C-CSS-20000 (d)
13. Press: ctrl-c To exit from driver	

14. From cdsbrowser: Highlight **/.:/RelA/Entry2** entry added	
15. From cdsbrowser: Press: Action Press: Delete Entry	Delete new entry using cdsbrowser

4.4.2.10 Test Case 10: Delete an Entry (TC011.010)

This test case must demonstrate that the Directory/Naming Service can delete an object Entry.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE/OODCE, Cell Directory Service, cdsbrowser, Clearcase, DNS

Data: Entry names with attributes and values.

Test Input:

Test driver main_hp or main_sun used to delete an object Entry.

Test Outputs

The object Entry is completely deleted.

Success Criteria

The Directory/Naming Service must demonstrate that the Entry can be successfully deleted, from the directory.

Test Procedures:

Test Case ID: TC011.010	
Test Name: Delete an Entry	
Test Steps:	Comments
1. Login to **client.**	
2. Perform DCE login. **"dce_login <username>"**	
3. Enter DCE **password**	
4. Enter: **cd /usr/testa/IT/tools/CSS/DN/<platform>**	substitute sun for hp, if testing sun platform
5. Enter: **cdsbrowser &**	Used to verify directory naming activity
6. Enter: **main_<**platform>	platform is hp or sun
7. From cdsbrowser: Press: **Tree** Press: **Update**	Refresh screen from cdsbrowser
8. Verify entry **/.:/RelA/Entry2** is displayed	
9. Monitor msgs for **Delete an Entry ...**	Press: Return until msg is displayed.
10. Verify entry **/.:/RelA/Entry2** is deleted.	C-CSS-20000 (a)
11. Press: ctrl-c To exit from driver	

4.4.2.11 Test Case 11: Duplicate Entry Names (TC011.011)

This test case will demonstrate that the Directory/Naming Service can detect any duplicate entry names being entered into the DB.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE/OODCE, Cell Directory Service, cdsbrowser, Clearcase, DNS

Data: Entry names with attributes and values.

Tools: N/A

Test Input:

Test driver main_hp or main_sun used to create entry.

Test Outputs

The Directory/Naming Service must reject any attempt to create a duplicate entry name in the CDS directory, along with sending an error message.

Success Criteria

A duplicate entry name is successfully rejected and an error message is sent, successfully.

Test Procedures:

Test Case ID: TC011.011	
Test Name: Duplicate Entry Names	
Test Steps:	Comments
1. Login to **client.**	
2. Perform DCE logins. **"dce_login <username>"**	
3. Enter DCE **password**	
4. Enter: **cd /usr/testa/IT/tools/CSS/DN**	substitute sun for hp, if testing sun platform
5. Enter: **cdsbrowser &**	Used to verify directory naming activity
6. From cdsbrowser: Press: **Action** Press: **Create Entry**	
7. From cdsbrowser: Create an entry name: **/.:/RelA/Entry2**	
8. From cdsbrowser: Verify entry name added: **/.:/RelA/Entry2**	
9. Enter: **main_<**platform>	platform is hp or sun
10. Monitor msgs for **Name /.:/RelA/Entry Loaded using EcDnCompositeName AddCtx method**	Press: Return until msg is displayed.
11. Verify that the **duplicate** entry name is rejected.	ERROR: Entry Already exists
12. List directory to verify entry name with elements was not added.	Verify from cdsbrowser.
13. Press: ctrl-c To exit from driver	

4.4.2.12 Test Case 12: Add Attribute to Non-Existant Element (TC011.012)

This test case must demonstrate that the Directory/Naming Service can detect a request to add an attribute to a non-existant element.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE/OODCE, Cell Directory Service, cdsbrowser, Clearcase, DNS

Data: Entry names with attributes and values.

Test Input:

Test driver main_hp or main_sun used to add an attribute to a non-existant element.

Test Outputs

Directory/Naming service sends the user notification that the element does not exist.

Success Criteria

The Directory/Naming Service must demonstrate the ability to detect non-existant elements by successfully sending some form of notification.

Test Procedures:

Test Case ID: TC011.0012	
Test Name: Add Attribute to Non-Existant Element	
Test Steps:	Comments
1. Login to **client.**	
2. Perform DCE login. **"dce_login <username>"**	
3. Enter DCE **password**	
4. Enter: **cd /usr/testa/IT/tools/CSS/DN/**	substitute sun for hp, if testing sun platform
5. Enter: **cdsbrowser &**	Used to verify directory naming activity
6. Enter: **main_<**platform>	platform is hp or sun
7. Monitor mgs for **Adding Element will set the Attribute...**	Press: Return until msg is displayed.
8. From cdsbrowser: **Double-click /.:/RelA/Entry2**	To access Attribute lists
9. From cdsbrowser: **Select attribute** Press: **REMOVE**	Removes Element (attribute/value)
10. From cdsbrowser: Verify Element (attributes/values) removed	
11. Monitor mgs for **Add more Values to the Element ...**	Press: Return until msg is displayed.
12. Monitor screen for any **ERROR messages or WARNINGS**	
13. Verify proper **Error notification** is received	ERROR: Attribute not found
14. Press: ctrl-c To exit from driver	

4.4.2.13 Test Case 13: Add Element to Non-Existant Entry (TC011.013)

This test case must demonstrate that the Directory/Naming Service can detect a request to add an element to a non-existant entry.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE/OODCE, Cell Directory Service, cdsbrowser, Clearcase, DNS

Data: Entry names with attributes and values.

Test Input:

Test driver main_hp or main_sun used to add an element to a Non-Existant Entry .

Test Outputs

Directory/Naming service sends the user notification that the entry does not exist.

Success Criteria

The Directory/Naming Service must demonstrate the ability to detect a non-existant entry by successfully sending some form of notification of the fact.

Test Procedures:

Test Case ID: TC011.0013	
Test Name: Add Element to Non-Existant Entry	
Test Steps:	Comments
1. Login to **client.**	
2. Perform DCE login. **"dce_login <username>"**	
3. Enter DCE **password**	
4. Enter: c setview <username>	Cleartool setview
5. Enter: **cd /usr/testa/IT/tools/CSS/DN/**	substitute sun for hp, if testing sun platform
6. Enter: **cdsbrowser &**	Used to verify directory naming activity
7. Enter: **main_<**platform>	platform is hp or sun
8. Monitor msgs **Name /.:/RelA/Entry2 Loaded using EcDnCompositeName AddCtx method**	Press: Return until msg is displayed.
9. From cdsbrowser: Press: **Tree** Press: **Update**	Refresh screen from cdsbrowser
10. Verify entry name added: **/.:/RelA/Entry2**	Verify value added using cdsbrowser
11. From cdsbrowser: Double-click on Entry name	Access Attribute lists
12. From cdsbrowser: **Select Action** Press: **Delete**	Deletes Entry
13. From cdsbrowser: Verify Entry deleted	
14. Monitor mgs for **Adding Element will set the Attribute...**	Press: Return until msg is displayed.
15. Monitor screen for any ERROR messages or WARNINGS	Driver attempts to ADD an element to non-existant Entry.
16. Verify proper ERROR message or Warning received.	ERROR: Entry does not exist
17. Press: ctrl-c To exit from driver	
18. From cdsbrowser: Highlight **/.:/RelA/Entry2** entry added	
19. From cdsbrowser: Press: Action Press: Delete Entry	Delete new entry using cdsbrowser

4.4.2.14 Test Case 14: Try to Read Deleted Values (TC011.014)

This test case must demonstrate that the Directory/Naming Service can detect an empty value list when trying to READ a deleted list of values.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE/OODCE, Cell Directory Service, cdsbrowser, Clearcase, DNS

Data: Entry names with attributes and values.

Tools: N/A

Test Input:

Test driver main_hp or main_sun used to get a value.

Test Outputs

Directory/Naming service sends notification to user on trying to READ an empty value list.

Success Criteria

The Directory/Naming Service must demonstrate that an attempt to READ a list of values deleted from the directory will send notification to the user, successfully.

Test Procedures:

Test Case ID: TC011.014	
Test Name: Try to Read Deleted Values	
Test Steps:	Comments
1. Login to **client.**	
2. Perform DCE login. **"dce_login <username>"**	
3. Enter DCE **password**	
4. Enter: **cd /usr/testa/IT/tools/CSS/DN/**	substitute sun for hp, if testing sun platform
5. Enter: **cdsbrowser &**	Used to verify directory naming activity
6. Enter: **main_<**platform>	platform is hp or sun
7. Monitor mgs for **Adding Element will set the Attribute...**	Press: Return until msg is displayed.
8. From cdsbrowser: **Double-click /.:/RelA/Entry2**	To access Attribute lists
9. From cdsbrowser: **Select attribute** Press: **REMOVE**	Removes values
10. Monitor mgs for **Getting Attribute/Value(s) ... Passing Attr/Value objects**	Press: Return until msg is displayed.
11. Monitor screen for any ERROR messages or WARNINGS	Driver attempts to Read deleted values
12. Verify proper ERROR message or Warning received.	ERROR: Empty
13. Press: ctrl-c To exit from driver	
14. From cdsbrowser: Highlight **/.:/RelA/Entry2** entry	
15. From cdsbrowser: Press: Action Press: Delete Entry	Delete new entry using cdsbrowser

4.4.2.15 Test Case 15: Try to Modify Non-Existant Value (TC011.015)

This test case must demonstrate that the Directory/Naming Service can detect a request to MODIFY a non-existant element value.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE/OODCE, Cell Directory Service, cdsbrowser, Clearcase, DNS

Data: Entry names with attributes and values.

Tools: N/A

Test Input:

Test driver main_hp or main_sun used to modify a value from an object element.

Test Outputs

The Directory/Naming Service sends notification to the user that the value requested to be modified, does not exist.

Success Criteria

The Directory/Naming Service must successfully send notification of some form that the value requested to be modified does not exist.

Test Procedures:

Test Case ID: TC011.015	
Test Name: Try to Modify Non-Existant Value	
Test Steps:	Comments
1. Login to **client.**	
2. Perform DCE login. **"dce_login <username>"**	
3. Enter DCE **password**	
4. Enter: **cd /usr/testa/IT/tools/CSS/DN/**	substitute sun for hp, if testing sun platform
5. Enter: **cdsbrowser &**	Used to verify directory naming activity
6. Enter: **main_<**platform>	platform is hp or sun
7. Monitor mgs for **Attribute Added Read...ReadElement**	Press: Return until msg is displayed.
8. From cdsbrowser: **Double-click /.:/RelA/Entry2**	To access Attribute lists
9. From cdsbrowser: **Select attribute** Press: **REMOVE**	Removes all values of the Attribute
10. Monitor mgs for **Modify a value of Attribute ...**	Press: Return until msg is displayed.
11. Monitor screen for any ERROR messages or WARNINGS	Driver attempts to Read deleted values
12. Verify proper ERROR message or Warning received.	ERROR: Empty
13. Press: ctrl-c To exit from driver	
14. From cdsbrowser: Highlight **/.:/RelA/Entry2** entry	
15. From cdsbrowser: Press: Action Press: Delete Entry	Delete new entry using cdsbrowser

4.4.2.16 Test Case 16: Duplicate Attributes (TC011.016)

This test case will demonstrate that the Directory/Naming Service can detect any duplicate Attributes being added to an entry.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE/OODCE, Cell Directory Service, cdsbrowser, Clearcase, DNS

Data: Entry names with attributes and values.

Tools: N/A

Test Input:

Test driver main_hp or main_sun used to add an Attribute to an object element.

Test Outputs

The Directory/Naming Service must reject any attempt to create a duplicate object element attribute in the CDS directory, along with sending an error message.

Success Criteria

A duplicate object element attribute is successfully detected and rejected with the proper error message received.

Test Procedures:

Test Case ID: TC011.016	
Test Name: Duplicate Attributes	
Test Steps:	Comments
1. Login to **client.**	
2. Perform DCE login. **"dce_login <username>"**	
3. Enter DCE **password**	
4. Enter: **cd /usr/testa/IT/tools/CSS/DN/**	substitute sun for hp, if testing sun platform
5. Enter: **cdsbrowser &**	Used to verify directory naming activity
6. Enter: **main_attr_dup**	main_attr creates duplicate attribute scenario
7. Monitor msgs **ADD DUPLICATE ATTRIBUTE NAME with new Values**	
8. Monitor screen for any ERROR messages or WARNINGS	
9. Verify proper notification received.	
10. Verify that **no** duplicate attribute names are added	
11. Press: ctrl-c To exit from driver	
12. From cdsbrowser: Highlight **/.:/RelA/Entry2** entry	
13. From cdsbrowser: Press: Action Press: Delete Entry	Delete new entry using cdsbrowser

4.4.2.17 Test Case 17: List Object Entries (TC011.017)

This test case will demonstrate that the Directory/Naming Service will list all object entry names when requested.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE/OODCE, Cell Directory Service, cdsbrowser, Clearcase, DNS

Data: Entry names with attributes and values.

Tools: N/A

Test Input:

Test driver main_hp or main_sun used to list object entries.

Test Outputs

The Directory/Naming Service lists all object entries.

Success Criteria

The Directory/Naming Service must completely list all object entries, successfully.

Test Procedures:

Test Case ID: TC011.017	
Test Name: List Object Entries	
Test Steps:	Comments
1. Login to **client.**	
2. Perform DCE login. **"dce_login <username>"**	
3. Enter DCE **password**	
4. Enter: **cd /usr/testa/IT/tools/CSS/DN/**	
5. Enter: **cdsbrowser &**	Used to verify directory naming activity
6. Enter: **main_hp** or main_sun for sun platform	main starts executable driver code
7. Enter: **x** x is used to step through driver code	
8. Monitor msgs **LISTING Context ... No of Entries**	
9. From cdsbrowser: Press: **Tree** Press: **Update**	Refresh screen from cdsbrowser
10. Verify number of Entries listed with number displayed on monitor.	C-CSS-20000 (b)
11. Press: ctrl-c To exit from driver	
12. From cdsbrowser: Highlight **/.:/RelA/Entry2** entry	
13. From cdsbrowser: Press: Action Press: Delete Entry	Delete entry using cdsbrowser

4.4.2.18 Test Case 18: Namespace Protection (TC011.018)

This test case will demonstrate that the Directory/Naming Service will reject any unauthorized access to the namespace.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE/OODCE, Cell Directory Service, cdsbrowser, Clearcase, DNS, acledit

Data: Entry names with attributes and values.

Tools: N/A

Test Input:

None

Test Outputs

The Directory/Naming Service must deny all improper access attempts by the user with the proper error notification.

Success Criteria

All improper access attempts are denied by Directory/Naming Service with proper error notification.

Test Procedures:

Test Case ID: TC011.018	
Test Name: Namespace Protection	
Test Steps:	Comments
1. Login to **client.**	
2. Perform DCE login. **"dce_login <username>"**	
3. Enter DCE **password**	
4. Enter: **cd /usr/testa/IT/tools/CSS/DN/**	substitute sun for hp, if testing sun platform
5. Enter: **cdsbrowser &**	Used to verify directory naming activity
6. Monitor msgs **Get LIST OF VALUES FOR ATTRIBUTE ...**	
7. From cdsbrowser: Press: **Tree** Press: **Update**	Refresh screen from cdsbrowser
8. From cdsbrowser: Verify entry name added: **/.:/RelA/Entry2**	
9. Using Acl Edit, Set ACL to Entry set to No permissions.	
10. Enter: **x** x is used to step through driver code	
11. Verify that permission is denied to **/.:/RelA/Entry2**	Verify proper notification received. C-CSS-20080, C-CSS-20085
12. Press: ctrl-c To exit from driver	
13. From cdsbrowser: Highlight new entry added	

4.4.2.19 Test Case 19: Namespace Replication (TC011.019)

This test case will demonstrate that the Directory/Naming Service provides a mechanism which periodically will update copies of the namespace from the namespace designated as the master via the skulk operation.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE, Cell Directory Service, cdsbrowser

Data: Entry names with attributes and values.

Tools: None

Test Input:

None

Test Outputs

The Directory Naming Service replicated server namespace(s) is identical to the master server namespace before and after any namespace updates to the master namespace.

Success Criteria

The Directory Naming Service replication server successfully maintains an identical namespace to the master server namespace.

Test Procedures:

Test Case ID: TC011.019	
Test Name: Namespace Replication	
Test Steps:	Comments
1. Login to master server.	
2. Login to replica server.	
3. Verify master and replication server namespaces are identical.	
4. Shutdown master server	
5. Update master namespace with new entry.	
6. Reactivate master namespace	
7 . Verify replica namespace is not changed	
8.Issue skulk command to update replica server namespace	
9. Verify replication server namespace automatically gets updated through skulking.	Replicated namespace must be identical to the namespace from the master namespace. C-CSS-20020, C-CSS-20030, C-CSS-20040

4.4.2.20 Test Case 20: X.500 Namespace Communication (TC011.020)

This test case will demonstrate that the Directory/Naming Service is capable of resolving Global Directory Service (GDS) namespace lookups via the NASA X.500 protocol communications.

Test Configuration:

Hardware: SunOS5.4, HP 9.05, X.500

Software: DCE/OODCE, Global Directory Service, cdsbrowser, GDA

Data: Entry names with attributes and values.

Tools: None

Test Input:

None

Test Outputs

The Directory/Naming Service resolves a global namespace lookup via the X.500 format.

Success Criteria

The Directory/Naming Service provides a successful global namespace lookup mechanism using the DCE intercell Global Directory Agent via the X.500 format.

Test Procedures:

Test Case ID: TC011.020	
Test Name: X.500 Namespace Communication	
Test Steps:	Comments
1. Login to local client in cell #1.	
2. Establish a remote intercell namespace lookup from cell #1 to cell #2 by using the x.500 GDS entry namespace convention.	C-CSS-20010, C-CSS-2020120, C-CSS-20130
3. Verify the lookup is resolved successfully.	
4. Attempt to establish a remote intercell namespace lookup from cell #1 to cell #2 by using a non-existent entry name.	
5. Verify the proper notification is received.	

4.4.2.21 Test Case 21: DNS Namespace Communication (TC011.021)

This test case will demonstrate that the Directory/Naming Service is capable of resolving Domain Name Service (DNS) namespace lookups.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: DCE/OODCE, Cell Directory Service, DNS

Data: Entry names with attributes and values.

Tools: None

Test Input:

None

Test Outputs

The Directory/Naming Service must deny all improper access attempts by the user with the proper error notification.

Success Criteria

All improper access attempts are denied by Directory/Naming Service with proper error notification.

Test Procedures:

Test Case ID: TC011.021	
Test Name: DNS Namespace Communication	
Test Steps:	Comments
1. Login to local client in cell #1.	
2. Establish a remote intercell namespace lookup from cell #1 to cell #2 by using the DNS entry namespace convention.	
3. Verify the lookup is resolved successfully.	C-CSS-20010, C-CSS-2020120, C-CSS-20130, C-CSS-20050
4. Attempt to establish a remote intercell namespace lookup from cell #1 to cell #2 by using a non-existent entry name.	
5. Verify the proper notification is received.	

4.4.3 Communication Infrastructure 1 Build (BC003)

The Communication Infrastructure 1 Build is an integration of the Message Passing 1 thread, the Directory Naming thread, and the DCE Security Authentication thread. The objective of this build is to verify that the previously tested functionality of these threads remains intact following integration.

4.4.3.1 Test Case 1: DCE Shared-Secret Key Authentication (BC003.001)

The purpose of this test is to create a new userid and password and to demonstrate the DCE shared-secret key authentication mechanism offered to clients by the server authentication protocols provided in an integrated basic communications test environment.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h, rgy_edit

Data: User login name and password

Tools: OODCE 1.0.3, RogueWave Tools.h++ 6.1

Test Input:

Inputs to this test case include valid username and password for DCE client login and client request for DCE shared-secret key authentication method.

Test Output:

The expected results of this test case is a successful authentication.

Success Criteria

This test will be deemed successful when the user is logged on successfully.

Test Procedures:

Build Test Case ID: BC003.001	
Test Name: DCE Shared-Secret Key Authentication	
Test Steps:	Comments
1. Login to <platform> workstation in **edfcell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
2. cleartool setview <username>	Setup user clearcase view
3. cd **/ecs/formal/CSS/REL_A**	
4. Enter: **source .buildrc**	
5. cd **/ecs/formal/CSS/REL_A/DOF/src/SEC/test**	
6. Enter: **clearmake -C gnu clean**	Cleanup old **header** and **object** files
7. Enter: **cp ServerMain.cxx.authn_secret ServerMain.cxx**	Set **server** software for RPC authentication **(rpc_c_authn_dce_secret)**
8. Enter: **clearmake -C gnu**	Compile software from clearcase
9. Enter: **cp SleeperClient.cxx.authn_secret SleeperClient.cxx**	Set **client** software for RPC authentication **(rpc_c_authn_dce_secret)**
10. Enter: **clearmake -C gnu**	Compile software from clearcase
11. Copy **ServerMain.cxx** and **SleeperClient.cxx** to your home directory.	
12. Login to <platform> workstation in **relacell**.	Platform is **sun** or **hp. Compile on the appropriate platform !**
13. cd **/usr/testa/IT/tools/CSS/SEC**	Location for test driver execution.
14. **ftp <workstation name> .hitc.com** ex. ftp casper.hitc.com	ftp to edfcell.
15. > **cd**	change directory to your home directory
16. > **ls**	List files in directory
17. > **bin**	binary mode

18. > **prompt**	disable prompting
19. > **mget S***	Transfers **Sleeper** files to the directory
20. Wait until all files are transferred	
21. > **quit**	exit ftp
22. Start application server. Enter: server &	Server will autolog itself and start listening.
23. Verify that server is started.	
24. Start application client. Enter **client**.	Client request authentication into server.
25. Verify client successfully logs on.	***C-CSS-21110***
26. Enter: **Kill -9 xxxx**	xxxx -> PID for **sleeper_server**
27. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	

4.4.3.2 Test Case 2: Transmission of Multiple Messages (BC003.002)

The purpose of this test is to verify that implementation two of the Message Passing Service is capable of transmitting multiple messages within an application integrated within the Communications Infrastructure Build 1. Specifically this test will show that an application may transmit messages concurrently by utilizing multiple instances of the message object.

Test Configuration:

Hardware: HP and Sun workstations/server HCI, Inter networking subsystem CI (LAN Network resources)-INCI

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions, ECS Common library functions.

Data: N/A

Tools: multi_cli_hp, multi_srv_hp, multi_cli_sun, multi_srv_sun.

Test Input:

A cds entry name must be supplied to the test client application. The user will be asked for the number of messages to send if it is not specified on the command line.

Test Output:

The client test driver should issue a statement indicating that the specified number of messages were sent and it is continuing its execution. The server test driver should issue statements indicating the time each message was received and processed. For each message the client will then display the transmission number, the uuid, the calculation performed, the time the message was received and processed by the server, and its return status.

Success Criteria:

The client application sends a message to the server and continues executing. The client receives return information from each message object indicating its transmission was successful.

Test Procedures:

Build Test Case ID: BC003.002	
Test Name: Transmission of Multiple Messages	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.
2. Create two sessions (i.e. open two xterm windows) and log into DCE in both by typing : **dce_login** *username password*	
3. In one window, start the cdsbrowser application in the background by typing: **cdsbrowser &**	Note the cdsbrowser application is available only on HP platforms. When executing this testcase on other platforms create a third session in step 2. In this window log into a HP machine in the same cell as the test machine, set your DISPLAY variable to point back to the test machine, and then execute step 3.
4. Start the test server application by typing the following in one of the test machine windows: **multi_srv_hp** *<cds entry name>*	The cds entry should be placed in a directory in which the executor of the test has write permission. For example, in the relacell the following is a valid entry name: */.:/RelA/msg_server* where */.:/RelA/* is the directory and *msg_server* is the server name. A message stating "The server is now listening...." should appear in the server window indicating that the server is running.
5. Verify the server object has been registered in the CDS by clicking the **tree->update** button in the cdsbrowser's toolbar.	The server object should now be displayed inside the user-specified directory (e.g. underneath the */.:/RelA/* directory)
6. In the second window start the client application by typing the following: **multi_cli_hp** *<cds entry name>*	The cds entry name should correspond to the cds entry name specified in step 4.
7. When prompted by the client test driver, enter the number of messages to transmit and press **Enter**.	

8. Verify the following message appears in the client window indicating the messages were issued: "*<#>* messages sent... continuing execution...."	The # sign should correspond to the number entered in step 7. The messages are sent using each object's EcDcDSyncCom::Send method. (C-CSS-22010) Each message object calls this function within a separate thread created by the test client application. The messages consist of two randomly generated numbers that are sent to the server where they are multiplied and their product is returned to the message object.
9. Verify a message similar to the following appears in the server window for each message sent: "Transmission *<transmission #>* received from client at *<time string>*"	The numbering of these messages may appear out of order. This is due to the fact that the transmission number corresponds to the order in which each message object's PreInvoke() method was first entered and not when the rpc call was actually made. The rpc call occurs within the Invoke() method.
10. Verify a message similar to the following appears in the server window for each message sent: "Transmission *<transmission #>* concluded processing at *<time string>*"	NOTE: Each rpc call sleeps during its execution for a distinct amount of time. The sleeping period is based upon its transmission number i.e. (transmission 1 will sleep for 1 second, transmission 2 will sleep for 2 seconds). This should, but not necessarily, cause the rpc's to finish processing in sequence.

11. Verify the following message appears in the client window for each message sent: "Message *<message #>* I am transmission: *<attempt #>* My uuid is: < *<returned uuid>>* I calculated: *<calculation string>* I was received on *<time string>* I was returned on *<time string>* Message's return status:1"	The message number corresponds to the order in which the message objects were created in the test driver. The transmission number corresponds to the order in which each message thread called the PreInvoke() function. Therefore these two numbers may not be the same. The indented portion of the message corresponds to the results retrieved from each message object via its GetResults() method. (C-CSS-22010). The UUID corresponds to the message object being transmitted. (C-CSS-22130). Normally the received time string will be identical for all the messages. However, if a large number (e.g. 50) of rpcs are sent to the same server at the same time then the server request queue may fill up and additional connections will be refused. A DCE object not found message should appear in this case for each failed rpc. In this driver, each message object will retry sending the rpc 10 times before considering that message transmission a failure. The message/thread return status is stored in the private member "_done" and can be obtained using the message object's Done() member function. A value of 1 indicates the thread completed successfully and -1 indicates the thread failed in some point of its execution. If the thread isn't complete a value of 0 will be returned. (C-CSS-22140)
12. Remove the server object from the cds namespace by highlighting the server object in the cdsbrowser and clicking on the **options->delete** entry button in the toolbar.	
13. In each window logout of dce by issuing the following commands: **kdestroy** **exit**	

4.4.3.3 Test Case 3: Directory Naming Capabilities (BC003.003)

The purpose of the Directory Naming Capabilities test is to demonstrate some of the functionality of the Directory Naming Service being integrated within the Communications Infrastructure Build 1. This functionality includes the ability to Create Entry Names into the CDS namespace by using the AddCtx method to add each context key or by loading a full context constructor string and to add/get the element to the object entry name.

Test Configuration:

Hardware: SunOS5.4, HP 9.05

Software: Cell Directory Service, cdsbrowser, DNS

Data: None

Tools: OODCE 1.0.3, RogueWave Tools.h++ 6.1

Test Input:

main_hp or main_sun used to create entry names using both AddCtx method and constructor to load full context string and to add/get elements.

Test Output:

The Directory Naming Service displays all entry names, including entries created and any elements added to the entry.

Success Criteria:

Entry names created in the namespace successfully display their names.

Test Procedures:

Build Test Case ID: BC003.003	
Test Name: Directory Naming Capabilities	
Test Steps:	Comments
1) Login to **client.**	
2) Perform DCE login. **"dce_login <username>"**	
3) Enter DCE **password**	
4) Enter: **cd /usr/testa/IT/tools/CSS/DN/<platform>**	substitute **sun** for hp, if testing sun platform
5) Enter: **cdsbrowser &**	Used to verify directory naming activity
6) Enter: **main_<**platform>	platform is **hp** or **sun**
7) Monitor msgs for **Name /.:/RelA/Entry2 Loaded using EcDnCompositeName AddCtx method.**	Press: **Return** until msg is displayed. AddCtx method loads each context key separately. **C-CSS-20000 (b)**
8) Press: **Tree** Press: **Update**	Refresh screen from cdsbrowser
9) Verify entry name added: **/.:/RelA/Entry2**	Verify Entry Name Created using cdsbrowser. **C-CSS-20000 (a)**
10) Monitor msgs for **Name /.:/RelA/test2 Loaded using EcDnCompositeName CONSTRUCTOR (a string.**	Press: **Return** until msg is displayed. CONSTRUCTOR method **C-CSS-20000 (b)**
11) Press: **Tree** Press: **Update**	Refresh screen from cdsbrowser
12) Verify entry name added: **/.:/RelA/test2**	Verify Entry Name Created using cdsbrowser. **C-CSS-20000 (a)**
13) Monitor msgs for **Adding Element will set the Attribute...**	Press: **Return** until msg is displayed. **C-CSS-20000 (c)**
14) From cdsbrowser: **Double-click /.:/RelA/Entry2**	To access Attribute lists
15) From cdsbrowser: **Select attribute added** Press: **Display**	Gets and displays Attribute(s)

16) Verify **Attribute** CSSAttr added	List Attribute **C-CSS-20000 (e)**
17) From cdsbrowser: **Double-click CSSAttr** to list values	Gets and displays an Attributes values
18) Verify added **Values** CSSValue0, 1, 2 are displayed	List Values **C-CSS-20000 (f)**
19) Press: ctrl-c To exit from driver	
20) From cdsbrowser: Highlight new entry **/.:/RelA/Entry2**	
21) From cdsbrowser: Press: Action Press: Delete Entry	Delete new entry using cdsbrowser

4.5 External Interface Communications Tests

The external interface communications tests consist of the following integrated threads and builds:

- User Account Management Thread (TC012)
- Kerberos ftp Thread (TC013)
- Kerberos Client/Server Thread (TC014)
- Security Authorization Thread (TC015)
- External Interface Communications Build (BC004)

4.5.1 User Account Management Thread (TC012)

The User Account Management Thread verifies the initial user registration function of the MSS Accountability Management Service. This function provides the capability to register one of two generic class of users : guest users and registered users. Guest users are those that have not formally requested to become registered users. Registered users are guest users that have submitted requests for a registered user account, and have had accounts created for them based on an approval process. User profiles are created from information provided by the users. These profiles contain user information that is shared with other subsystems to perform functions such as E-Mail notification and product distribution. New user accounts are created whenever a new record is added to the user profile database.

4.5.1.1 Test Case 1: User Registration Request (TC012.001)

This test case verifies the capability to submit a request as a guest user for a registered user account. This request is stored in a Sybase database of pending requests for user accounts. Each request contains the information needed to create a user profile record. New user accounts are created whenever a new record is added to the user profile database.

This test does not verify operator access to pending requests for user requests for user accounts, nor is there automatic registration of users from the pending request database. However, user profiles are created and stored in the user profile database, and new user accounts are created.

Test Configuration:

Hardware: MSS Server (HP)

Software: MsAcUsrProfile, MsAcUsrReqMgr, MsAcGuestUserReg

Data: Sample user information, runserver.csh/runclient.csh drivers

Tools: DCE (rgy_edit), Sybase

Test Input:

Runserver.csh driver inserting user profile data into Sybase.

Test Output:

Sybase will return successful creation and insertion into database.

Success Criteria:

This test is successful if a guest user can enter the appropriate user registration information and be accepted as a registered user.

Test Procedures:

Test Case ID: TC012.001	
Test Name: User Registration Request	
Test Steps:	Comments:
1. Login to the HP machine where the MsAcUsrProfile resides.	Successful login.
2. Set your terminal environment display enter: **setenv DISPLAY <***host ip address***>:0.0**	
3. Create a keytab file for the executables. From the command line on the HP, enter: **dce_login <***username***>**	
4. Enter password for the username.	
5. Enter **rgy_edit <return>**	
6. At the rgy_edit prompt, enter: >ktadd -p cell_admin -f MsAcUsrProfileServer.keytab	
7. Enter password for username and quit	
8. Enter **kdestroy** while in the dce mode.	
9. Enter exit to get out of dce.	
10.From the HP command line, enter: **cd /home/kcampbel** **runserver.csh** (driver will enable the Profile Server)	The runserver.csh enables the Profile Server.
11.The following messages will appear: • **Server is running, DB is ready** • **Registered User server is ready**	
12.Enter **ps** to verify that the MsAcServer is running.	
13.Enter **runclient.csh** (driver will add users into MsAcUsrProfile	

14.The runclient.csh driver will do the following: • **Insert Request** • **Retrieve Request List** • **Create Registered User** • **After creating registered user, it will Delete Request** • **List Registered Users** • **List User Profiles** • **List Daily, Weekly, Monthly Usage**	The MSS Accountability Management Service shall create a new user account whenever a new record is added to the user profile database. C-MSS-75020
15.Enter **ps** to find and kill the MsAcServer process, enter: **kill -9 <process #>**	
16.End test	

4.5.1.2 Test Case 2: Verify User Profile (TC012.002)

This test case verifies the capability to verify the user profile Sybase database that is maintained that includes the following information about a registered user:

- Name
- User-ID
- Telephone number
- E-Mail address
- Organization

Test Configuration:

Hardware: MSS Server (HP)

Software: MsAcGuestUserReg, MsAcUsrProfile, MsAcPrincipal

Data: Sample user information submitted with the runserver.csh driver

Tools: Sybase

Test Input:

User generated data.

Test Output:

User generated data.

Success Criteria:

This test is successful if the user profile information is returned from the Sybase queries.

Test Procedures:

Test Case ID: TC012.002	
Test Name: Verify User Profile	
Test Steps:	Comments:
1. Login into The HP where the MsAcGuestUserReg resides.	Successful login.
2. Set your terminal environment display enter: **setenv DISPLAY <***host ip address***>:0.0**	
3. From the command line on msse4hp, enter: **source /vendor/sybase/*.csh**	This enables the Sybase environment..
4. From the command line on the HP, enter: **isql -U prof_db_role**	
5. Enter **password** for prof_db_role	
6. At the Sybase promt, enter: • **sp_help** • **go** **This will list the tables in Sybase**	Sybase tables will be listed.
7. At the Sybase promt, enter: **select * from MsAcUsrProfile** This will list the user profiles that were entered in Sybase.	The MSS accountability shall provide the capability to maintain a user profile database that stores the following information: a. Name b. User ID c. Telephone Number d. E-mail address e. Organization C-MSS-75000
8. At the Sybase promt, enter: **exit** (to exit out of Sybase)	
9.End test	

4.5.2 FTP/Kerberos FTP Thread (TC013)

File Transfer Protocol (FTP) is used to send and receive files on a network. A program is run on the client machine which connects to a process on a server machine. The client may then send and receive files when this connection is established.

Kerberos is a network authentication system for use on physically insecure networks. It allows entities communicating over networks to prove their identity to each other while preventing eavesdropping or false identity attacks. It also provides data stream integrity (detection of modification) and secrecy (preventing unauthorized reading) using cryptography systems such as DES (data encryption standard). Kerberos works by providing principals (users or services) with tickets that they can use to identify themselves to other principals and secret cryptographic keys for secure communication with other principals. A ticket is a sequence of a few hundred bytes that can be embedded in a network protocol. thereby allowing the processes implementing that protocol to validate the identity of the principals involved.

This thread verifies file transfer and file management capabilities through interactive access and application interfaces across a distributed environment. Kerberos FTP is tested by verifying the ability to transfer files across networks while preserving data integrity and privacy. Testing will consist of:

- sending/receiving files interactively and in batch mode
- sending/receiving files between remote hosts
- sending/receiving files with Kerberos encrypted data

4.5.2.1 Test Case 1: Interactive File Transfer (TC013.001)

The purpose of the this test is to verify that the CSS File Access Service provides basic connection oriented operations for interactive file transfers using the file transfer protocol (FTP). FTP allows read access to all users. This test verifies the capability of sending and receiving both ASCII and binary files interactively and non-interactively between hosts. This test also verifies that existing files are not overwritten if files with same name are requested for transfer.

Test Configuration:

Hardware: SUN Solaris 2.4, HPUX 9.05, and SGI:IRIX 5.3

Software: ftp

Data: ASCII/Binary files representative of the science products available in Release A

Tools: None

Test Input:

Interactive ASCII or binary file transfer to a simulated local and remote DAAC. Verify file integrity and that no files were overwritten. File transfer to SCF host (infrastructure) via ftp. Verify file transfers between the mini-DAAC and the infrastructure.

Test Output:

Successful connection to both local and remote DAAC hosts. Complete file transfers interactively from the source to the destination successfully without lost data. Matched file sizes for files at both the sending and receiving sites. Appropriate time/date stamps. No error messages should be displayed.

Success Criteria:

This test is successful if files are sent and received without errors, valid file size match, checksums are valid, and files are not overwritten.

Test Procedures:

Test Case ID: TC013.001	
Test Name: Interactive File Transfer	
Test Steps:	Comments:
1. Login to local DAAC host	

2. Type: **setenv DISPLAY <**host IP address or name**>:0.0**	Setting visualization for the user host ex. setenv DISPLAY 155.157.123.25:0.0
3. Perform a ftp command to a local DAAC host. Enter: **ftp** <*hostname*> Provide a username and a password.	If an authorized user name and password are valid, then successful permission to enter the file transfer program is generated. The "*open"* command could also be used to establish connections while in ftp mode Satisfies C-CSS-60520, C-CSS-60600
4. Check the local status: Type: **status**	Identifies the *local* connection host, mode, verbose status, store and receive status, case status, number of transmission status, number of default file name mappings, printing status, proxy connection if any, and case status
5. Verify the file transfer efficiency statistics is "on". Type: **verbose;** type again to turn "on" if status says "off "	Default is "on" Provides authentication to file transfer efficiency statistics locally and remotely satisfies: C-CSS-60920
6. To ensure that the user does not overwrite a file (s) locally or remotely, check the current settings: (step 5) Type **runique**: for files being received Type **sunique**: for files being sent	No overwritten files via ftp (sent or received) Default is "off "
7. Type **status**	Verify the status is "on" for both runique and sunique
8. Select the file(s) to transfer locally. Files of type ASCII or Binary; **get** *filename* (one) **mget** *filename* (multiple)	Default mode is ASCII File (s) are transferred to the current directory
9. View output listing: **!ls -al** Verify transfer by matching file sizes	A listing of new file(s) transferred View local local directory contents satisfying: C-CSS-60510
10. Create an error scenario: can not capture a file (s): **get <** invalid file with protect permissions or non-existent file**>**	Error message saying no such file or directory
11. Change transfer mode to binary: Type **binary**	Response: Type set to I satisfies: C-CSS-60610
12. Repeat step 8 before continuing to step 13 - using binary file	
13. **!ls -al**	verification of binary transferred file(s) satisfying: C-CSS-60510
14. Set mode back to the default (ASCII) Type: **ASCII**	Response: Type set to A satisfies: C-CSS-60610

15. Select the file(s) to transfer remotely **put** or **send** *filename* (one) **mput** *filename* (multiple)	File (s) from the user's current directory is sent to a remote host
16. **!ls -al**	The initial state should show a received file in its current directory
17. Change transfer mode to binary: Type: **binary**	
18. Repeat step 15,16	Binary files are sent satisfies: C-CSS-60500
19. Enter: **!ls -al**	verification of binary transferred file(s)
20. Change Transfer mode is (**ASCII)**	Default: ASCII satisfies: C-CSS-60500
21. Initiate a close: **close**	The ftp session is closed
22. 'Open' a session: **open <**hostname**>**	Establishes a new ftp session Provide user login name and password
23. Again, initiate a file transfer of ASCII or Binary type; See steps 8 and 9	File transfer verification is performed to indicate time stamps (length of time for the transfer), checksums, and file sizes
24. Type: **close**	closes a ftp client session
25. Type: **bye**	Exits ftp

4.5.2.2 Test Case 2: Batch Schedule File Transfer (TC013.002)

The purpose of the this test is to verify that the CSS Access Service will provide an API to use the CsFtFTPScheduObj object to schedule ASCII and binary file transfer in batch mode from one remote workstation to a local workstation using the CsFtFTPRelA object. The test will include initializing a non-interactive operation, and verifying that the results of the operation are logged in operator specified log files.

Test Configuration:

Hardware: SUN(s) Solaris 2.4, HPUX(s) 9.05

Software: CsFtFTPSchedObj, CsftpFTPRelA.cxx, server.cxx, MyMangedServer.h, MyManagedServer.cxx, client.cxx, testclient.cxx, MsM.idl, MsM.H, MsM.cxx, sleeper.idl, sleeper.cxx, CliMessage.cxx, CliMessage.h, CliMesssageCb.h, CliMesssageCb.cxx, common.h

Data: ASCII/Binary files representative of the science products available in Release A

Tools: runscript

Test Input:

Schedule generation of multiple file transfers which are sent to a simulated DAAC host. Files can be of ASCII or binary type. This test will initialize a non-interactive operation and log the results to an operator specified file, including error messages if a scheduled operation fails.

Test Output:

Expected results will include entries to the operator specified log file with valid time-stamps and transfer completion messages. The operator must verify alarms sent and the events performed if a schedule operation fails.

Success Criteria:

This test is successful if the CSS file access service has demonstrated a schedule file transfer in batch mode and event logging was performed as needed.

Test Procedures:

Test Case ID: TC013.002	
Test Name: Batch File Transfer	
Test Steps:	Comments:
1. Login to a local DAAC host	Valid password needed
2. If necessary, type: **setenv DISPLAY <host IP address or name>:0.0**	Setting visualization for the user host ex. setenv DISPLAY 155.157.123.25:0.0
3. **mkdir** <data location>	'Log 'directory will be holding area for files to be transferred at a later time
4. **cd** <created data location>	Verify creation
5. Create subdirectories for <data location>: **mkdir** file1 file2 file3 file4 file5	
6. Execute the application: **./runscript**	Events are generated if a scheduled operation fails satisfies: C-CSS-60800 satisfies: C-CSS-60820
7. Verify the application logged to specified data location: **cd** < data location> Perform a '**more'** <data location> or the '**cat** ' <data location> commands to view the data files	Verify if the data was dumped into the correct locations satisfies: C-CSS-60810 satisfies: C-CSS-60820
8. Perform an ASCII or Binary file transfer: **ftp** <hostname>	User name and password provided
9. Attempt to extract a non-existent file in ftp mode : **get** <non-existent file > or <protected file>	Error message saying no such file or directory or privilege denied by the requested user
10. Exit ftp mode: '**bye**' or '**quit**'	
11. Review the event logger and analyze the data	Once the queue is full, the next attempted transfer will fail due to the maximum number of transfers has reached its limit satisfies C-CSS-60820

4.5.2.3 Test Case 3: Remote File Access (TC013.003)

The purpose of the this test is to verify that the CSS File Access Service can support anonymous file transfer, proxy mode transfer, and remote file transfer via ftp protocol. The remote files must be accessed as if they were part of the local file system. This includes the capability to list remote files either by complete file name or by specifying wildcards.

Test Configuration:

Hardware: SUN Solaris 2.4, HPUX 9.05, and SGI:IRIX 5.3

Software: CsftFTPRelA.cxx, CsFtFTPRelA.h, ftptest.C

Data: ASCII/Binary files representative of the science products available in Release A

Tools: OODCE 1.0, RogueWave Tools.h++ 6.1.0 or version 5.17

Test Input:

Local access to a DAAC host followed by a remote access within the same DAAC. User must be able to access files remotely and authenticate the functions of the FTP client package. Privileges and access controls must be requested and verified for authorization.

Test Output:

Remote access from a local DAAC is accepted for usage and operations.

Success Criteria:

This test is successful if remote files can be accessed via ftp. Also, the anonymous ftp and proxy mode must be supported by the 'ftp' program. All file transfers must be time stamped and verified with accurate size and contents of the file transferred from the remote host.

Test Procedures:

Test Case ID: TC013.003	
Test Name: Remote File Access/Proxy	
Test Steps:	Comments:
1. Login to local DAAC host.	User name and password needed
2. Type: **cd** /ecs/formal/CSS/REL_A/DOF/FTP/ftp View long listings: **ls -al**	
3. Perform a file transfer to a host outside the local host domain **ftp** <hostname>	
4. Gather status and set file transfer statistics: **status** **runique** **sunique**	Current default setting for some ftp options

5. Close ftp session: **close**	
6. Initiate a remote ftp: **open** <hostname>	User name and password needed
7. Perform **get** <remote file local file >	A copy of the remote file from the remote machine is transferred
8. Type: **!ls -al**	Verify the ASCII file transfer by matching the check sums
9. Perform **put <** local file remote file>	A copy of the local file is sent to the remote terminal satisfies: C-CSS-60640
10. Type: **!ls -al**	
11. Type: **ls** **status**	list contents of the remote directory satisfies: C-CSS-60630 Reads: No proxy connection
12. Type: **proxy open** <hostname> Type: **status**	Send a command to an alternate connection Status should say there is a proxy connection to host terminal specified satisfies: C-CSS-60620
13. Type: **status**	Status of the remote machine indicates a proxy connection
14. Type: **bye**	Ends both the original ftp and the remote proxy connection
15. Perform anonymous file transfer: **ftp** <hostname>	Current anonymous users have to use newsroom.hitc.com in EDF
16. Type: **anonymous (user name) password: email address**	Guest login with limited access
17. Type: **ls -F**	List of files with the denoted '/'
18. Perform: **get <**filename>	Attempt to get a file
19. Type: !**ls -al**	Verify receipt of a file. Match checksums
20. Perform: **put** or **send** filename	Attempt to place or send a file. Error message should appear; anonymous users can not place files at sites
21. Type: **quit**	Exit ftp session

4.5.2.4 Test Case 4: Sending/Receiving Kerberos Encrypted Files (TC013.004)

The purpose of the this test is to verify that kerberos software is capable of sending and receiving files with kerberos encrypted data between local and remote terminals via kftp. Also, the test verifies the appropriate imposition of access privilege for security control.

Test Configuration:

Hardware: SUN Solaris 2.4, HPUX 9.05, and SGI:IRIX 5.3

Software: Kerberos COTS, CsFtFTPRelA

Data: ASCII/ text data files

Tools: Network analyzer: Sniffer; encryption tool

Test Input:

Through the Kerberos application, a test data file is encrypted and transmitted across the FDDI network. The file transfer attempts are made using the kerberos/kftp software, and the data file must be encrypted. The network analyzer attempts to access the encrypted data file, reviewed its contents, and verify a transfer between the terminals in use.

Test Output:

The network analyzer accesses the encrypted data file and displays a summary of the data file.

Success Criteria:

This test is successful if the files are transferred between the local and remote hosts via kftp, and valid users obtain the information contained in the data files.

Test Procedures:

Test Case ID: TC013.004	
Test Name: Sending and Receiving Kerberos	
Test Steps:	Comments:
1. Login into distributed client/server environment (DCE): **dce_login** <principal name>	Authorized users will be permitted Password required
2. Go to the kerberos directory: **cd /**krb5/bin/kftpdir	Users can login into the daemon if they have a kerberos ticket to prove their identity
3. Enter: **kinit** < *user name*>; provide password	The DCE server maintains a database of user, server, and password information. Kinit command creates your credentials cache and stores your credentials in it Master database contains entries for all network services that require Kerberos authentication
4. Check service keys existence: **cd** /etc/v5srvtab	Services are only ran by **root.**
5. Check to see if the environment variable: 'KRB5CCNAME' is set	Access is allowed if you are in the /etc/passwd file; otherwise entry is denied.
6. To list credentials : **klist** To destroy kerberos credentials: **kdestroy**	Hence: 'kftpdir' is usually found in /krb5/bin Remember to kdestroy after completing all entries performed in the distributed environment
7. Perform a kftp command: **kftp** < *host name*> or if no path set : **./kftp**	If an authorized user name, then successful permission to enter the file transfer program Hence: **No password** needed if the user authenticated. A MESSAGE SIGNIFIES A TICKET HAS BEEN APPROVED satisfies C-CSS-60900 C-CSS-60920

8. Verify the file transfer efficiency statistics is "on" . Type: **verbose;** type again to turn "on" if status says "off "	Default is "on" Provides authentication to file transfer efficiency statistics locally and remotely
9. Check the local status; Type: **status**	
10. To ensure that the user does not overwrite a file (s) locally or remotely, check the current settings: (step 10) Type **runique**: for files being received Type **sunique**: for files being sent	No overwritten files via ftp (sent or received)
11. Type: **status**	Verify the status is "on" for both runique and sunique entries
12. Select the file(s) to transferred **get** < filename>(one) **mget** <filename> (multiple)	File (s) are transferred to the current directory
13. Enter: **!ls -al**	Verify check sums
14. Enter: **bye**	Exits ftp mode
15. Enter: **kdestroy**	Remove credentials
16. Use the network analyzer to capture the 'kftp' command sent across the network -Select FDDI Analyzer -Start data capture/filter -Perform a ' kftp' command from an alternate terminal -End data capture and search for the sent command in the network data capture on screen or save to file - View what must appear to be encrypted data	
17. Attempt another connection: **kftp** <hostname>; if authenticated no password required	Verify connection using network analyzer
18. Enter: **kdestroy**	
19. Attempt to perform the 'kftp' with an invalid user: **kftp** <*host name*> Waiting for password; no credentials provided and user is unable to use this application	No authentication
20. Repeat step # 19 with another invalid user	No authentication

4.5.3 Client/Server Gateways Thread (TC014)

The client/server environment is a software partitioning paradigm in which a distributed system is split between one or more server tasks which accept requests, according to some protocol, from (distributed) client tasks, asking for information or action. There may be either one centralized server or several distributed ones. This model allows clients and servers to be placed independently on nodes in a network.

The purpose of this thread is to verify client/server process for two communication gateways: TRMM Science Data and Information System (TSDIS) and Science Data Processing Facility (SDPF). The operational scenario for TSDIS is as follows:

a) TSDIS sends a data availability notice (DAN) to ECS to archive. The specifications will include: data file names, file dates/times, number of files, and file locations.

b) The DAN is validated by ECS and a data availability acknowledgment (DAA) is returned.

c) TSDIS utilizes the kerberos ftp application to transfer the files. Kerberos is a network authentication system for use on physically insecure networks. It allows entities communicating over networks to prove their identities to each other.

d) Files are verified against the DAN and logged in the data delivery notice (DDN).

e) ECS sends TSDIS the DDN reporting log status of errors if any associated with the transferred files. Successfully completed files are ingested and archived.

f) TSDIS responds with a data delivery acknowledgment (DDA).

g) For any failed transfer, another DAN is sent after corrections.

h) For network failure during or before transfers, ECS retransmits, but no retransmission is required from TSDIS

The operational scenario for SDPF is as follows:

a) ECS retrieves data from SDPF to archive.

b) SDPF connects the ECS gateway on the specified port.

c) SDPF send a DAN message specifying file information.

d) ECS validates the DAN, and sends a corresponding DAA.

e) Files are transferred via ftp. FTP is a program/protocol to transfer and receive files across the network. Files are also ingested and archived.

f) ECS send SDPF a DDN containing file transfer results.

g) SDPF responds to ECS with a corresponding message, DDA.

In addition, this thread verifies the combination of the TSDIS and SDPF gateways as they relate to the client/server process. The combination scenario is as follows:

a) ECS retrieves data from SDPF.

b) Gateway receives socket call from the port from SDPF.

c) Authentication Request message arrives.

d) Gateway authenticates the client using user identification and password. A authentication response is sent back

e) TSDIS sends a DAN message.

f) Gateway makes a remote procedure call (RPC) to Ingest server and gets a DAA.

g) Gateway sends the DAA to SDPF.

h) Ingest retrieves the data specified in the DAN and makes a RPC to Gateway with DDN.

i) Gateway sends the DDN to SDPF.

j) SDPF send the Gateway the corresponding DDA.

Also, this thread consists of testing privacy between the client/server interface when verifying client privileges, passing data, and requesting service utilization.

In conclusion, this thread verifies the APIs provided for data transfer and management services, electronic messaging services, remote terminal services, process to process communication services, directory and user access control services , network management services, network security, access control services, internetwork interface services, and the bulletin board service.

4.5.3.1 Test Case 1: Verify TSDIS Communications (TC014.001)

This test verifies the TSDIS communications can successfully transfer data files to ECS over a configurable port utilizing the incoming and delivery gateways. Files are transferred via kftp within the distributed computing environment (DCE) cell configuration. This test must demonstrate the capability of the TSDIS gateway to:

- perform protocol conversion between socket messages and OODCE remote procedure calls, listening at configurable port, and place all messages in network byte order
- utilized kerberos identity through encryption, provide data integrity, and authenticate messages through the DCE/Kerberos service
- route requests received based on initialization information, log communication errors, log authentication requests, and receive information from external entities and DCE servers
- exit the process upon receiving a shutdown message from the internal ECS server (incoming gateway)

Test Configuration:

Hardware: SUN workstation, Gateway server, Ingest server, Data server, Kerberos security server

Software: CsGwMessage.cxx, CsGwSdpfMsg.cxx, CsGwGateway.cxx, CsGwIncomingGatewayMain.cxx, CsGwDeliveryGatewayMain.cxx, CsGwIntGatewaymain.cxx, CsGwDlvGatewaymain.cxx, CsGwSdpfMain.cxx

Data: configuration file, keytab file, data files via kftp

Tools: Network monitoring tool: Sniffer, Kerberos 5 beta 5, OODCE 1.0.3, Rogue Wave tools.h++, net.h++ 6.04

Test Input:

Users must change the configuration file to reflect the incoming and delivery gateway executables. A configurable port is set for TSDIS gateway.

Test Output:

Data files are transferred to ECS via kftp. The data server places the data files in specified locations. The incoming gateway validates with a DAA if no errors found, and the delivery gateway sends a DDA for successfully completed data transfers files.

Success Criteria:

This test will be successful if the external client can connect to a configurable port and verify DAN -> DAA incoming and DDN -> DDA delivery gateways. Also, security considerations must be met.

Test Procedures:

Test Case ID: TC014.001	
Test Name: Verify TSDIS Communications	
Test Steps:	Comments:
1. Start the network monitoring tool to monitor communication across the line.	
2. As a client, login to a DCE Cell, **dce_login** <principal name>; password	
3. From the monitoring tool, attempt to read information that the server is reading from the client such as: a. client's id b. client's permissions c. client's group d. client's memory capability	
4. Set the port connection through FDDI switch	

5. Start client session via kftp: **cd /krb5/bin/kftpdir** **kinit** <principal name>; password **klist** **kftp** <host>; username and password	Go the application directory to run “kftp”, and gather credential tickets for authentication
6. Gather authentication response from external process	satisfies: C-CSS-64040, C-CSS-64080, C-CSS-64090
7. Send a DAN to internal process and to the gateway in RPC	
8. Verify a DAA was returned	
9. Verify data files were verified against the DAN an logged in the DDN.	
10. Demonstrate an incomplete data transfer by sending another DAN with corrections inserted	satisfies: C-CSS-64060
11. Perform **kdestroy**	End DCE/kerberos session
12. Execute these: **dce_login** <principal name>; password **kinit** < username> password **klist**	
13. Execute: **. /** **CsGwDeliveryGatewayMain**	Test driver in window 1(delivery of mesages)
14. Execute: **. / tisserver** with port number	Test driver in window 2 (TSDIS server: received a DDN and send out a DDA)
15. Execute: **. / testClient**	Test driver in window 3(simulate Ingest’s sending program: sending a DDN and receiving DDA) satisfies: C-CSS-64020, C-CSS-64030
16. Incoming gateway test, execute**:** **. / tgclient**	
17. Execute a failure in the transferring of the file via kftp.	
18. TSDIS sends a new DAN with the corrections	
19. In case of network failure, no retransmissions necessary because TSDIS sends a new DAN with the corrections.	TSDIS does not retransmit in case of fail file transfers
20. To remove credentials: **kdestroy**	
21. Verify files: **cd /krb5/test/** **cat stdout** **cd /krb5/test/** **cat stderr**	Check log for errors
	satisfies: C-CSS-64010, C-CSS-64100, C-CSS-64120
22. Disconnect the server connection.	
23. Verify the server disconnected. ps -aceg |grep server	satisfies: C-CSS-64120

4.5.3.2 Test Case 2: Verify CSS SDPF Communications (TC014.002)

This test verifies SDPF communications can successfully transfer data files to ECS over a configurable port utilizing a constantly running gateway server that transport messages using the file transfer protocol (ftp). Files are transferred via ftp within the DCE cell configuration. This test must demonstrate the capability of the SDPF gateway to:

- perform protocol conversion between socket messages and OODCE remote procedure calls, listening at configurable port, and place in network byte order all messages
- route requests received based on initialization information, log communication errors, log authentication requests, and receive information from external entities and DCE servers
- exit the process upon receiving a shutdown message from the internal ECS server (incoming gateway)

Test Configuration:

Hardware: SUN workstation, Gateway server, Ingest server, Data server

Software: CsGwMessage.cxx, CsGwSdpfMsg.cxx, CsGwGateway.cxx, CsGwIncomingGatewayMain.cxx, CsGwDeliveryGatewayMain.cxx, CsGwIntGatewaymain.cxx, CsGwDlvGatewaymain.cxx, CsGwSdpfMain.cxx

Data: configuration file, keytab file, data files via ftp

Tools: Network monitoring tool: Sniffer, Rogue Wave tools.h++, net.h++ 6.04

Test Input:

TSDIS makes a socket connection on a configurable port to the the ECS Gateway or RPC connections on a configurable port.

Test Output:

ECS retrieves the data specified via ftp, logs the data to a DDN, awaits a DDA from SDPF.

Success Criteria:

This test will be successful when port connections are configured and messages are transferred successfully to SDPF.

Test Procedures:

Test Case ID: TC014.002	
Test Name: Verify CSS SDPF Communications	
Test Steps:	Comments:
1. Start the network monitoring tool to monitor communication across the line.	
2. As a client, login to a DCE Cell: **dce_login** <principal name>; password **klist**	
3. Open a TCP/IP connection on a specified port	
4. A authentication request is performed by SDPF client to the CSS-SDPF Gateway. Verify authentication request through DCE authorization.	
5. Verify configuration file is set for SDPF.	Satisfies: C-CSS-64070
6. Start the gateway main program with the configuration file name.	
7. Execute: **ingest_recv_DAN**	simulation of internal Ingest server to receive the outgoing DAN message (gateway for listening to a configurable port)
8. Execute: **ingest_send_DDN**	simulation of internal Ingest server to send out the DDN message (DDA is also received)
9. Execute: **sdpfClient**	sdpf client unit test simulation of internal Ingest server to send out the DDN message (requires client with server name, port number, and DAN_file_name)
10. Verify the SDPF gateway will make rpcs to Ingest Server with the messages and receive the response message from the rpc.	satisfies: C-CSS-64000, C-CSS-64110, C-CSS-64100, C-CSS-64010
11. Make sure the gateway is always running, status checks: **ps -aceg |grep sdpfmain**	
12. Verify the DDN has the file transfer results.	Satisfies: C-CSS-64020, C-CSS-64030
13. Logout of the DCE Cell and disconnect the server connection. **Kdestroy**	kills credentials: DCE/kerberos
14. Verify the gateway exits if non-authenticated responses are received and shutdowns upon receiving a 'shutdown control command'.	satisfies: C-CSS-64060, C-CSS-64120

4.5.4 Security Authorization Thread (TC015)

Authorization is the process of identifying what users/groups should be granted or denied access to any services or resources. In authorization, each resource is associated with a set of permissions given to different kinds of users for different types of access operations. This is used to selectively grant or deny certain principals access to specific resources. Authorization is performed by Access Control List (ACL) mechanisms. An ACL is an entry with information such as the name of the user/group and an associated permission schema which indicates the type of permissions given for the user/group.

4.5.4.1 Test Case 1: Name-Based Authorization (TC015.001)

The purpose of this test is to demonstrate that a client requesting RPC processing on a principle name authorization basis only, can get the name-based authorization servicing by a server defined for name-based authorization.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: Supplied by user input

Tools: OODCE 1.0.3, RogueWave Tools.h++ 6.1, BuilderXcessory 3.5.1p

Test Input:

Inputs include client and server binaries Sleeper_Server_NB and Sleeper_Client_NB, along with shell scripts server_NB, client_NB, and client_NB_err.

Test Output:

The expected results of this test case are a successful name-based authorization.

Success Criteria

This test will be deemed successful when a name-based authorization defined client and name-based authorization defined server, successfully processes RPC request from the client.

Test Procedures

Test Case ID: TC015.001	
Test Name: Name-Based Authorization	
Test Steps:	Comments
1. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory for server.
2. Start application server. Enter: **server_NB &**	Server will autolog itself and start listening.
3. Login to remote client.	
4. Perform DCE login. **"dce_login <username>"**	
5. Enter DCE **<password>**	
6. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory on client.
7. Enter: **client_NB_err**	Client shell script attempts Name-Based authorization with invalid client principle name.
8. Verify server API rejects an invalid client principle.	ERROR: Sleeper_Server: Caller not authorized to perform remote_sleep1.
9. Enter **client_NB**	Client shell script attempts Name-Based authorization with valid client principle name.
10. Verify RPC executed with Name-Based authorization successfully.	C-CSS-21120

11. Enter: **Kill -9 <Process ID>**	
12. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	

4.5.4.2 Test Case 2: User with Individual Permissions (TC015.002)

The purpose of this test case is to demonstrate that an authorized user within the DCE cell is restricted to access services only according to his own permission set.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

ACL Entry Editor (GUI)

Data: Supplied by user input

Tools: Sybase 10.01, OODCE 1.0.3, RogueWave Tools.h++ 6.1, BuilderXcessory 3.5.1p

Test Input:

Inputs to this test case include valid username and password for DCE account with individual (not group) access permissions.

Test Output:

The expected results of this test include successful login to the DCE cell, execution of valid services, and rejection of unassigned services.

Success Criteria:

This test will be deemed successful when the user securely logs into the DCE cell and executes all services assigned within the access control list. Success also depends on denial to execute services not assigned within the access control list.

Test Procedures

Test Case ID: TC015.002	
Test Name: User with Individual Permissions	
Test Steps:	Comments
1. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory for server.
2. Start server. Enter: **server &**	
3. Verify that server is listening.	
4. Login to client.	
5. Perform DCE login. **"dce_login** <username>**"**	
6. Enter DCE **<password>**	
7. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory on client.

8. Invoke ACL Entry Editor.	GUI
9. Enter ACL Name: <principle name>	Get a printable representation of the permissions C-CSS-21160 (c) Identifies the permissions available to a principle C-CSS-21160 (a)
10. Select Entry Type: **user**	To set permissions for user
11. Select Entry Permissions: **x, c**	xc -> Allows execution and permission assignment control
12. Press: **Add**	Adds permissions for user
13. Press: **Apply**	Commits Entries to database
14. Start client driver. Enter **client**.	Client will login into server.
15. Verify RPC request successfully processed.	
16. Invoke ACL Entry Editor.	GUI
17. Enter ACL Name: <Principle Name>	Same name as in step 9.
18. Select Entry Type: **user**	To set permissions for user
19. Select Entry Permissions: **x**	x -> Allows execution
20. Press: **Delete**	Deletes x permission for user and removes user execution privilege
21. Press: **Apply**	Commits Entries to database
22. Start client driver. Enter **client**.	Client will login into server.
23. Verify **RPC** request from user is successfully rejected	
24. Enter: **Kill -9 <Process ID>**	
25. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	

4.5.4.3 Test Case 3: Test Calling Principle Permissions (TC015.003)

The purpose of this test case is to demonstrate by using the acledit API a calling principles permissions can be tested as to whether they are granted or denied.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: Supplied by user input

Tools: OODCE 1.0.3, RogueWave Tools.h++ 6.1, BuilderXcessory 3.5.1p

Test Input:

Inputs to this test case include valid username and password for DCE account with individual access permissions.

Test Output:

The expected results of this test are that the permissions displayed are the permissions that are granted the principle.

Success Criteria:

This test will be deemed complete when each of the calling principle permissions are successfully tested as to whether they are granted or denied to the calling principle.

Test Procedures

Test Case ID: TC015.003	
Test Name: Test Calling Principle Permissions	
Test Steps:	Comments
1. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory for server.
2. Start server. Enter: **server &**	
3. Verify that server is listening.	
4. Login to client.	
5. Perform DCE login. **"dce_login <username>"**	
6. Enter DCE **<password>**	
7. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory on client.
8. Enter: **acledit** Invoke ACL editor.	
9. Enter: **pe**	Displays permissions implemented for the object by the objects ACL manager.
10. Enter: **g**.	Display calling principle permissions
11. Enter: **t** <**permission token**>	Test (t) each of the valid permissions against the display for the calling principle permissions separately for whether the permission is granted or denied. Token example: r r is for read access C-CSS-21160 (g) satisfied
12. Verify calling principle permission listed is either granted or denied according what's displayed for the calling principle.	
13. Enter: **exit**	Exit acledit
14. Enter: **Kill -9 <Process ID>**	

4.5.4.4 Test Case 4: Users with Single Group Permission (TC015.004)

The purpose of this test case is to demonstrate that an authorized user within the DCE cell is restricted to access services within his group's Access Control List.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: Supplied by user input

Tools: Sybase 10.01, OODCE 1.0.3, RogueWave Tools.h++ 6.1, BuilderXcessory 3.5.1p

Test Input:

Inputs to this test case include various combinations of valid user names, valid passwords, and the permissions assigned to the corresponding user logins.

Test Outputs

The expected results are successful login to the DCE client and activity execution corresponds to restrictions within the access control list.

Success Criteria

The test will be deemed successful when the user securely accesses the DCE client and all activity executions correspond to the access permissions within the access control list. In addition, all restricted access permissions will be verified.

Test Procedures

Test Case ID: TC015.004	
Test Name: Users with Single Group Permissions	
Test Steps:	Comments
1. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory for server.
2. Start server. Enter: **server &**	
3. Verify that server is listening.	
4. Login to client.	
5. Perform DCE login. **"dce_login <username>"**	
6. Enter DCE **<password>**	
7. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory on client.
8. Invoke ACL Entry Editor.	GUI
9. Enter ACL Name: <principle name>	
10. Select Entry Type: **user**	To set permissions for user
11. Select Entry Permission: **c**	c -> Allows permission assignment control
12. Press: **Add**	Adds c permission for user
13. Select Entry Type: **group**	To set permissions for specific principle group. Groupname must match Kerberos ticket group names
14. Highlight specific groupname from acl display	
15. Select Entry Permission: **c**	c -> Allows permission assignment control
16. Verify only permission c displays through Entry Key window.	
17. Press: **Add**	Adds c permission for users group
18. Press: **Apply**	Commit updates to Acl database
19. Start client driver. Enter **client**.	Client will login into server.
20. Verify RPC request from user is successfully rejected .	RPC cannot successfully processed without x permission.
21. Invoke ACL Entry Editor.	GUI
22. Enter ACL Name: <principle name>	
23. Select Entry Type: **group**	To set permissions for specific principle group. Groupname must match Kerberos ticket group names

24. Highlight specific groupname from acl display	same as in step 14
25. Select Entry Permission: **x**	x -> Allows execution
26. Verify only permissions c and x displays through Entry Key window.	Note: user permissions remain as c, while group permissions have c and x.
27. Press: **Add**	Adds x permission to principles group
28. Press: **Apply**	Commit updates to Acl database
29. Start client driver. Enter **client**.	Client will login into server.
30. Verify RPC request from principle is successfully processed.	
31. Enter: **Kill -9 <Process ID>**	
32. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	

4.5.4.5 Test Case 5: Users with Multi-Group Permissions (TC015.005)

The purpose of this test case is to verify that a user can be given authorization permissions from more than one group. The authorization permissions for the user are the union of the permissions from all groups to which the user belongs.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

ACL Entry Editor

Data: Supplied by user input

Tools: Sybase 10.01, OODCE 1.0.3, RogueWave Tools.h++ 6.1, BuilderXcessory 3.5.1p

Test Input:

Input to this test case includes a valid user name and password of a DCE account belonging to more than one ACL group.

Test Outputs

The expected results of this test include the successful assignment of multiple groups to any single user.

Success Criteria

This test will be deemed successful when a single user is assigned multiple groups and gains access to the services that each group offers exclusively, independent of any other user or group permission assignments.

Test Procedures

Test Case ID: TC015.005	
Test Name: Users with Multi Group Permissions	
Test Steps:	Comments
1. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory for server.
2. Start server. Enter: **server &**	
3. Verify that server is listening.	
4. Login to client.	
5. Perform DCE login. **"dce_login <username>"**	
6. Enter DCE **<password>**	
7. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory on client.
8. Invoke ACL Entry Editor.	GUI
9. Enter ACL Name: <principle name>	
10. Select Entry Type: **group**	To set permissions for specific principle group. Groupname must match Kerberos ticket group names
11. Highlight specific groupname_1 from acl display	
12. Select Entry Permission: **c**	c -> Allows permission assignment control
13. Verify only permission c displays through Entry Key window.	
14. Press: **Add**	Adds c permission for users group
15. Highlight specific groupname_2 from acl display	
16. Select Entry Permission: **x**	x -> Allows execution
17. Verify only permission x displays through Entry Key window.	
18. Press: **Add**	Adds x permission for users group
19. Press: **Apply**	Commit updates to Acl database
20. Select Entry Type: **user**	To set permissions for user
21. Select all Entry Permissions.	
22. Press: **Delete**	Delete all user defined permissions
23. Start client driver. Enter **client**.	Client will login into server.
24. Verify RPC request from user is successfully processed.	RPC successfully processed with groupname_2 x permission. This verifies that the execute permission is granted to the principle via the groupname_2 ACL assignment.
25. Enter ACL Name: <other principle name>	Enter a different principle name
26. Select Entry Type: **user**	To set permissions for specific principle group. Groupname must match Kerberos ticket group names
27. Select all Entry Permissions listed.	
28. Press: **Delete**	Deletes permissions to other user
29. Select Entry Permission: **x, r, w**	xrw -> Allows execution, read and write permissions.
30. Verify only permissions x, r, and w displays through Entry Key window.	
31. Press: **Add**	Adds permissions to other user
32. Press: **Apply**	Commit updates to Acl database
33. Verify that the update to the other user ACLs is successful.	This verifies that the control permission is granted to the principle via the groupname_1 ACL assignments.

34. Enter: **Kill -9 <Process ID>**	
35. Delete server entry via cdsbrowser 36. **</.:/cdsDirNam/cdsServerEntryName>**	

4.5.4.6 Test Case 6: User with ACL Modify Permission (TC015.006)

The purpose of this test case is to verify that a user with "C", create permissions, is able to modify the ACL. The test also verifies that the CSS Security Service is capable of logging audit information into security logs whenever authentication and authorization services are used.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: Supplied by user input

Tools: Sybase 10.01, OODCE 1.0.3, RogueWave Tools.h++ 6.1, BuilderXcessory 3.5.1p

Test Input:

Inputs to this test include a valid user name and password of a DCE account belonging to a group with create permissions.

Test Outputs

The expected results of this test include successful login to the DCE cell, execution of ACL_Edit, and modification of group permissions.

Success Criteria

This test will be deemed successful when the user securely logs into the DCE cell and successfully edits group permissions within ACL_Edit. These edits will be verified by accessing the "client" to check new access permissions.

Test Procedures

Test Case ID: TC015.006	
Test Name: User with ACL Modify Permissions	
Test Steps:	Comments
1. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory for server.
2. Start server. Enter: **server &**	
3. Verify that server is listening.	
4. Login to client.	
5. Perform DCE login. **"dce_login <username>"**	
6. Enter DCE **<password>**	
7. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory on client.
8. Invoke ACL Entry Editor.	GUI

9. Enter ACL Name: <principle name>	
10. Select Entry Type: **user**	To set permissions for user
11. Select Entry Permission: **c**	c -> Allows permission assignment control
12. Verify only permission c displays through Entry Key window.	
13. Press: **Add**	Adds c permission for user
14. Press: **Apply**	Commit updates to Acl database
15. Enter ACL Name: <other principle name>	Enter a different principle name
16. Select Entry Type: **user**	To set permissions for specific principle group. Groupname must match Kerberos ticket group names
17. Select all Entry Permissions listed.	
18. Press: **Delete**	Deletes permissions to other user
19. Select Entry Permission: **x, r, w**	xrw -> Allows execution, read and write permissions.
20. Verify only permissions x, r, and w displays through Entry Key window.	
21. Press: **Add**	Adds permissions to other user
22. Press: **Apply**	Commit updates to Acl database
23. Verifies that the update to the other user ACLs is successful.	This verifies that the control permission is granted to the principle.
24. Enter: **Kill -9 <Process ID>**	
25. Delete server entry via cdsbrowser 26. **</.:/cdsDirNam/cdsServerEntryName>**	

4.5.4.7 Test Case 7: User without ACL Modify Permission (TC015.007)

The purpose of this test case is to verify that an unauthorized user (one without create privileges) is unable to edit/modify the ACL.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: Supplied by user input

Tools: Sybase 10.01, OODCE 1.0.3, RogueWave Tools.h++ 6.1, BuilderXcessory 3.5.1p

Test Input:

Input to this test include a valid user name and password of a DCE account without individual or group create permission.

Test Outputs

The expected results of this test include successful login to the DCE cell, execution of ACL_Edit, and rejection to privilege modifications.

Success Criteria

This test will be deemed successful when the user securely logs into the DCE cell and is rejected from editing the ACL. The ACL will be monitored by accessing the "client".

Test Procedures

Test Case ID: TC015.007	
Test Name: User without ACL Modify Permissions	
Test Steps:	Comments
1. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory for server.
2. Start server. Enter: **server &**	
3. Verify that server is listening.	
4. Login to client.	
5. Perform DCE login. **"dce_login <username>"**	
6. Enter DCE **<password>**	
7. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory on client.
8. Invoke ACL Entry Editor.	GUI
9. Enter ACL Name: <principle name>	
10. Select Entry Type: **user**	To set permissions for user
11. Select Entry Permission: **c**	c -> Allows permission assignment control
12. Press: **Delete**	Deletes permission to user
13. Press: **Apply**	Commit updates to Acl database
14. Enter ACL Name: <other user name>	Enter a different principle name
15. Select Entry Type: **user**	To set permissions for other user
16. Select all Entry Permissions listed.	
17. Press: **Delete**	Deletes permissions to other user
18. Select Entry Permission: **x, r, w**	xrw -> Allows execution, read and write permissions.
19. Verify only permissions x, r, and w displays through Entry Key window.	
20. Press: **Add**	Adds permissions to other user
21. Press: **Apply**	Attempt to commit updates to Acl database
22. Verify modification attempt is rejected.	Error msg: Not authorized to modify (Valid msg)
23. Press: Cancel	
24. Invoke ACL Entry Editor.	GUI
25. Enter ACL Name: <other principle name>	same as step 14
26. Verify that there are no changes to other principle.	
27. Enter: **Kill -9 <Process ID>**	
28. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	

4.5.4.8 Test Case 8: Authenticate User Before Checking Authorization (TC015.008)

The purpose of this test case is to demonstrate that a user must get authenticated before checking whether the user is authorized to access a service or resource.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: Supplied by user input

Tools: OODCE 1.0.3, RogueWave Tools.h++ 6.1, BuilderXcessory 3.5.1p

Test Input:

Inputs to this test case include valid username and an invalid password for DCE principle account.

Test Output:

The expected results of this test are that the server and/or principle are authenticated before checking a principles authorization to access services and/or resources.

Success Criteria:

This test will be deemed successful when a server and/or principle are authenticated before checking a principles authorization to access services and/or resources.

Test Procedures

Test Case ID: TC015.008	
Test Name: Authenticate User Before Checking Authorization	
Test Steps:	Comments
1. cd **/usr/testa/IT/tools/CSS/SEC**	Change directory for server.
2. Enter: **server_no_pswd &**	Server script command file with no server password. Server not authenticated. ERROR: you must enter -cdsName <name> -princName <name> -keyFile <file> on command line
3. Login to client.	
4. Perform DCE login. **"dce_login <username>"**	
5. Enter DCE **<password>**	

6. Invoke ACL Entry Editor.	ERROR: Warning - binding to ACL's server is unauthenticated Warning - binding to registry is unauthenticated ERROR: authentication service invalid (DCE / sec) Unable to bind to object /.:/RelA/EcSeSecurityServer/RemoteSleepObj1
7. Verify no authorization without authentication of server and principle.	C-CSS-21110
8. Enter: **Kill -9 <Process ID>**	
9. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	

4.5.4.9 Test Case 9: CSS to MSS Login Notification (TC015.009)

The purpose of this test case is to demonstrate the capability of the CSS Security Service to notify the MSS Management Agent service upon a predetermined number of unsuccessful login attempts.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: Supplied by user input

Tools: OODCE 1.0.3, RogueWave Tools.h++ 6.1, BuilderXcessory 3.5.1p

Test Input:

Input to this test include the EcSeLogin_3 executable.

Test Outputs

The expected results of this test include notification to the MSS Management Agent upon a predetermined number of unsuccessful login attempts.

Success Criteria

This test will be deemed successful when a predetermined number of unsuccessful login attempts successfully notify the MSS Management Agent service.

Test Procedures

Test Case ID: TC015.009	
Test Name: CSS To MSS Login Notification	
Test Steps:	Comments

1. Enter: **cd /usr/testa/IT/tools/CSS/SEC**	change directory
2. Enter: **klist**	Verify credentials ticket not created
3. Enter: **EcSeLogin_3**	Display Usage instructions
4. Enter: **EcSeLogin_3 <username>**	Enter Valid Username
5. Enter Password: **<password>**	Enter Valid password
6. Enter: klist	Verify credentials ticket created
7. Enter: kdestroy	Destroy Ticket
8. Enter: **EcSeLogin_3 <username>**	Enter Valid Username
9. Enter Password: **<password>**	Enter Invalid Password; Repeat 2 times. Only 3 attempts are allowed.
10. Verify **Exit** from Password prompt.	
11. Enter: **klist**	Verify credentials ticket not created
12. Verify DCE login attempts notify MSS Agent.	C-CSS-21105
13. Enter: **EcSeLogin_3 <username>**	Enter Invalid Username.
14. Verify Invalid username rejected by DCE	ERROR: User ID Failure: Registry Object Not found (DCE/sec)
15. Enter: **EcSeLogin_3 <username> <password>**	Enter Valid username and Valid password on one line.
16. Enter: **klist**	Verify credentials ticket created
17. Enter: kdestroy	Destroy Ticket
18. Enter: **EcSeLogin_3 <username> <password>**	Enter Valid username and Invalid password on one line
19. Enter: **klist**	Verify credentials ticket created
20. Enter: **kdestroy**	Destroy Ticket
21. Verify DCE login attempts notify MSS Agent.	C-CSS-21105

4.5.4.10 Test Case 10: Permission Schema Definition (TC015.010)

The purpose of this test case is to demonstrate that the CSS Security Service provides an API to define the permission schema associated with a server or resource.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Schema Editor

Data: Schema set

Tools: Sybase 10.01, OODCE 1.0.3, RogueWave Tools.h++ 6.1, BuilderXcessory 3.5.1p

Test Input:

Supplied by user input

Test Outputs

The expected results of this test include demonstrating the ability to define additional permission schema to the schema provided by DCE.

Success Criteria

This test will be deemed when additional permissions are successfully defined to the existing schema provided.

Test Procedures

Test Case ID: TC015.010	
Test Name: Permission Schema Definition	
Test Steps:	Comments
1. Enter: cd /u**sr/testa/IT/tools/CSS/SEC**	Change directory for server.
2. Start server. Enter: **server_perm &**	
3. Verify that server is listening.	Verify default permission schema.
4. Enter: **pe**	List current permissions available.
5. Login to client.	
6. Perform DCE login. **"dce_login <username>"**	
7. Enter DCE **<password>**	
8. Enter: cd /u**sr/testa/IT/tools/CSS/SEC**	Change directory on client.
9. Invoke Schema Editor.	Bringup Schema Editor screen.
10. Enter: <**ACL DB Name>**	
11. Enter: **<Application Name>**	
12. Enter: **A**	Permission Token string 'A'. Token character 'c' cannot be used.
13. Enter: **RelA_I&T_Token_Test**	Type new permission usage definition.
14. Press: **Add**	Add new permission token.
15. Press: **Apply**	Apply the new permission to ACL database.
16. Verify list of permissions reflects additional 'A' permission added	See Schema Editor menu
17. Invoke ACL Entry Editor	Bringup ACL Entry editor screen
18. Verify new permission is displayed in the list of available ACLs	
19. Enter: <**ACL Name**>	Displayed in the Entry Key view
20. Select ACL resource to give permissions	
21. Select **Single** for Entry Type	
22. Select Entry Permissions: **c, x, r, w, A**	Select new permission with some standard permissions.
23. Press: **Add**	
24. Verify selected permissions are displayed from the Acl Entry Editor menu.	
25. Press: **Apply**	Apply the permissions to ACL database.
26. Bringup acledit	
27. Enter: **pe**	List current ACL schema
28. Verify new permission token displayed.	
29. From Acl Schema Editor screen, again bringup ACL DB Name:	ACL DB name from step 11
30. Select the '**w**' token (write permission)	

31. Press: **change**	
32. Change w to **B**	B is a user defined permission
33. Press: **Apply**	Apply the change to the ACL database
34. Verify the change from the ACL Entry Editor display.	See Entry Permission list
35. From acledit Enter: **pe**	List current ACL Schema
36. Verify permission **B** is displayed.	
37. From Acl Schema Editor screen, again bringup ACL DB Name:	ACL DB name from step 11
38. Select the '**B**' token.	B is a user defined permission
39. Press: **Delete**	
40. Press: **Apply**	
41. Verify B permission deleted from the ACL Entry Editor display.	
42. From acledit Enter: **pe**	
43. Verify permission **B** is not displayed.	
44. Enter: **Kill -9 <Process ID>**	
45.	C-CSS-21130

4.5.4.11 Test Case 11: ACL Database (TC015.011)

The purpose of this test case is to demonstrate the capability of the CSS Security Service to create and maintain (Add, Change, Delete) any prescribed ACLs associated with any server or resources in a database.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

ACL Editor

Data: Supplied by user input.

Tools: Sybase 10.01, OODCE 1.0.3, RogueWave Tools.h++ 6.1, BuilderXcessory 3.5.1p

Test Input:

Supplied by user input

Test Outputs

The results demonstrate that CSS Security services is capable of creating and maintaining an ACL database.

Success Criteria

This test will be deemed successful when the CSS Security services successfully create and maintain an ACL database with the proper permissions set for each resource.

Test Procedures

Test Case ID: TC015.011	
Test Name: ACL Database	
Test Steps:	Comments
1. Enter: cd /u**sr/testa/IT/tools/CSS/SEC**	Change directory for server.
2. Start server. Enter: **server_AclDB &**	
3. Verify that server is listening.	
4. Invoke ACL Editor	
5. Select ACLDB from menubar	
6. Create ACLDB name **ACLDBI&T**	New ACL database created
7. Verify new ACLDB displays from ACL Editor screen	
8. Invoke Schema Editor	
9. Enter ACL DB Name: **ACLDBI&T**	
10. Verify permission schema is defaulted for new ACL DB	All new ACL databases must initially be set for the default permission schema.
11. Press: **Apply**	ACL database created.
12. Invoke ACL Entry Editor	Used to assign permissions to each resource
13. Enter Principle or Resource Name: <ACL Name>	
14. Select Entry Type: **Single**	
15. Select Entry Permissions allowed.	Assignment of permissions for resource.
16. Press: **Add**	
17. Verify that the assigned ACLs display from screen	
18. Press: **Apply**	Apply permissions to ACL database
19. From ACL Editor, display ACL database and the acls assigned to the database.	
20. Verify acls assigned are added as members of the ACL database.	
21. Select the Entry Permission token '**r**' (read).	
22. Press: **Change**	
23. Change the read to another permission.	
24. Verify change is displayed on menu screen.	
25. Press: **Apply**	Apply change to ACL database.
26. Select the Entry Permission token '**w**' (write)	
27. Press: **Delete**	Delete the write permission from the ACL database.
28. Verify 'w' write permission is not displayed on menu screen.	
29. Press: **Apply**	Apply deletion to ACL database.
30. Press: **Cancel**	
31. Verify all modifications completed from ACL Entry Editor display.	Maintenance complete for single resource.
32. Enter: **Kill -9 <Process ID>**	
	C-CSS-21140

4.5.4.12 Test Case 12: Multiple ACL Databases (TC015.012)

The purpose of this test case is to demonstrate the capability of the CSS Security Service to create/delete any multiple of ACL databases and ACL schema associated with the main ACL database.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Schema Editor, ACL Editor, ACL Entry Editor

Data: Supplied by user input

Tools: Sybase 10.01, OODCE 1.0.3, RogueWave Tools.h++ 6.1, BuilderXcessory 3.5.1p

Test Input:

Inputs include client and server binaries Sleeper_Server_Acl2DB and Sleeper_Client_Acl2DB, along with shell scripts server_Acl2DB, client_Acl2DB.

Test Outputs

The expected results of this test demonstrate the capability to create multiple ACL databases to the ACL database repository.

Success Criteria

This test will be deemed successful when multiple ACL databases are successfully created by the APIs provided for the CSS Security service.

Test Procedures

Test Case ID: TC015.012	
Test Name: Create Multiple Database	
Test Steps:	Comments
1. Enter: cd /u**sr/testa/IT/tools/CSS/SEC**	Change directory for server.
2. Start server. Enter: **server_Acl2DB &**	
3. Verify that server is listening.	
4. Invoke ACL Editor	
5. Create new ACL database named **AclDB_1**	
6. Invoke Schema Editor	
7. Create AclDB_1 schema with **c, r, w, x, t, d, J, K, L** permissions.	
8. Verify schema is assigned to AclDB_1 only.	
9. Create another ACL database named **AclDB_2**	
10. Create AclDB_2 schema with **c, r, w, x, A, B** permissions.	
11. Verify schema is assigned to AclDB_2 only.	

12. Create the ACL database acl resource members for **AclDB_1** with the following permissions: **GG1 : c, r, w, x, t, d, J, K, L** **GG2 : c, r, x, d, J, K** **GG3 : r, x, K, L** **GG4 : J, K, L** **GG5 : L** ***EC6 : c, r, A, B** *Error condition	AclDB_1 schema contains the Unix, DCE, and User Defined permissions as part of the overall ACL schema.
13. Verify selected permissions are successfully assigned to AclDB_1 database, except for EC6 resource.	
14. Verify proper error condition is generated for EC6 resource.	
15. Create the ACL database acl resource members for **AclDB_2** with the following permissions: **HH1 : c, r, w, x, A, B** **HH2 : c, r, w, A, B** **HH3 : r, w, x** **HH4 : A, B** **HH5 : A** !**GG1 : c, r, w, x, A, B** !same name as in AclDB_1 database *!**GG2 : c, r, x, d, J, K** *Error condition	AclDB_2 schema contains the Unix, DCE, and User Defined permissions as part of the overall ACL schema.
16. Verify selected permissions are successfully assigned to AclDB_2 database, except for GG2 resource.	
17. Verify proper error condition is generated for GG2 resource.	
18. From AclDB_1, **change** permission J to A.	
19. Verify for AclDB_1 change is successful from ACL Schema screen, and had no effect on AclDB_2 schema.	
20. From AclDB_1, **delete** the 'w' permission from the ACL schema.	
21. Refresh screen and verify for AclDB_1 change is successful from ACL Schema screen, and had no effect on AclDB_2 schema.	
22. Verify acl resource permissions reflect the updates from the schema.	
	C-CSS-21140

4.5.4.13 Test Case 13: ACL Persistent Database Store (TC015.013)

The purpose of this test case is to demonstrate the capability of the CSS Security Service to save and retrieve the ACL database onto persistent storage.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

ACL Editor, ACL Entry Editor

Data: Supplied by user input.

Tools: Sybase 10.01, OODCE 1.0.3, RogueWave Tools.h++ 6.1, BuilderXcessory 3.5.1p

Test Input:

Supplied by user input

Test Outputs

The expected results of this test demonstrate that the in the event of the loss of the ACL database server, CSS Security provides recovery of the ACL database and maintains the original ACL database settings.

Success Criteria

This test will be deemed successful when, after the recovery of the ACL database the original ACL database values remain as originally set.

Test Procedures

Test Case ID: TC015.013	
Test Name: ACL Persistent Database Store	
Test Steps:	Comments
1. Enter: cd /u**sr/testa/IT/tools/CSS/SEC**	Change directory for server.
2. Start server. Enter: **server_AclDB &**	
3. Verify that server is listening.	
4. Invoke ACL Editor.	
5. Create a new ACL database using the ACL Editor.	
6. Create 5 new acl resource members using the ACL Entry Editor.	Save acls onto persistent storage.
7. Verify ACL database created with 5 acl members.	Verify acls from ACL Editor display
8. Enter: **Kill -9 <Process ID>**	Simulated server SHUTDOWN.
9. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	
10. Restart server. Enter: **server_AclDB &**	Simulated server REBOOT.
11. Verify that server is listening.	
12. Invoke ACL Editor.	
13. Verify ACL database and 5 acl resource members persist.	Retrieve acls from persistent storage.
14. Enter: **Kill -9 <Process ID>**	
	C-CSS-21150

4.5.4.14 Test Case 14: CSS Security Auditing (TC015.014)

The purpose of this test case is to demonstrate the capability of the CSS Security Service to log audit information into security logs whenever authentication or authorization services are used.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: Supplied by user input

Tools: OODCE 1.0.3, RogueWave Tools.h++ 6.1, BuilderXcessory 3.5.1p

Test Input:

Inputs include client and server binaries Sleeper_Server_Audit and Sleeper_Client_Audit, along with shell scripts server_Audit and client_Audit.

Test Outputs

The expected results include a record logged into the security audit log which specifically pertains to a principles authentication and authorization attempts. The information contained in the record logged must include the following :

Date and Time of event

User Name

Type of event

Success or failure of the event

Origin of the request

Success Criteria

This test is deemed successful when the proper event information is logged into the security audit log.

Test Procedures

Test Case ID: TC015.014	
Test Name: CSS Security Auditing	
Test Steps:	Comments
1. Enter: **cd /usr/testa/IT/tools/CSS/SEC**	Change directory for server.
2. Start server. Enter: **server &**	server application name
3. Verify that server is listening.	
4. Login to client.	

5. Perform DCE login. **"dce_login <username>"**	
6. Enter DCE **<password>**	
7. cd /usr/testa/IT/tools/CSS/SEC	Change directory on client.
8. Enter: **client**	client application name
9. Verify **authentication** attempt gets logged to MSS Agent security audit log with required audit information.	C-CSS-21210
10. Verify **authorization** attempt gets logged to MSS Agent security audit log with required audit information.	C-CSS-21210
11. Enter: **Kill -9 <Process ID>**	
12. Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	C-CSS-21150

4.5.4.15 Test Case 15: Test Another Principle Permissions (TC015.015)

The purpose of this test case is to demonstrate the capability of the CSS Security Service to test if another principle, other than the calling principle, has some permissions.

This test case satisfies requirement C-CSS-21160(h) through inspection. The rdacl_ routine rdacl_test_access_on_behalf is a COTS routine provided by OSF DCE. This rdacl_() interface is directly accessible to a routine called sec_acl_(). The calls beginning with sec_acl are used by a client program that wishes to access the ACLs. An ACL_Edit program provided by OSF uses this interface, as do ACL management programs provided by vendors. The sever actually does not implement the sec_acl_ routines. The sec_acl routines bind to stub routines provided by DCE. They in turn call routines that begin with rdacl_, and it is these routines that the ACL manager implements.

Reference: OSF DCE Guide to Developing Distributed Applications,
Harold W. Lockhart, Jr.
Chapter 11 sections 11.2.2 and 11.2.3,
Chapter 22 sections 22.2 and 22.4.9

4.5.4.16 Test Case 16: Identify ACL Mgr Protecting Object (TC015.016)

The purpose of this test case is to demonstrate the capability of the CSS Security Service to identify ACL managers protecting objects, by returning all supported manager UUIDs.

This test case satisfies requirement C-CSS-21160(b) through inspection. The rdacl_ routine rdacl_get_manager_types is a COTS routine provided by OSF DCE. This rdacl_() interface is directly accessible to a routine called sec_acl_(). The calls beginning with sec_acl are used by a client program that wishes to access the ACLs. An ACL_Edit program provided by OSF uses this interface, as do ACL management programs provided by vendors. The sever actually does not implement the sec_acl_ routines. The sec_acl routines bind to stub routines provided by DCE. They in turn call routines that begin with rdacl_, and it is these routines that the ACL manager implements.

Reference: OSF DCE Guide to Developing Distributed Applications, Harold W. Lockhart, Jr.
Chapter 11 sections 11.2.2 and 11.2.3,
Chapter 22 sections 22.2 and 22.4.10

4.5.4.17 Test Case 17: Locate Writeable Copy of ACL (TC015.017)

The purpose of this test case is to demonstrate the capability of the CSS Security Service to be able to locate the server with the writeable copy of the ACL, by returning a reference to the manager instance that can perform updates.

This test case satisfies requirement C-CSS-21160(d) through inspection. The rdacl_ routine rdacl_get_referral is a COTS routine provided by OSF DCE. This rdacl_() interface is directly accessible to a routine called sec_acl_(). The calls beginning with sec_acl are used by a client program that wishes to access the ACLs. An ACL_Edit program provided by OSF uses this interface, as do ACL management programs provided by vendors. The sever actually does not implement the sec_acl_ routines. The sec_acl routines bind to stub routines provided by DCE. They in turn call routines that begin with rdacl_, and it is these routines that the ACL manager implements.

Reference: OSF DCE Guide to Developing Distributed Applications, Harold W.Lockhart, Jr
Chapter 11 sections 11.2.2 and 11.2.3,
Chapter 22 sections 22.2 and 22.4.13

4.5.5 External Interface Communications Build (BC004)

The External Interfaces Communications Build is an integration of the User Account Management, FTP/kFTP, Gateway Client/Server, and Security Authorization threads. The functions to be tested consist of some of the key functionality's of these threads.

The purpose of this build is to integrate these threads into a comprehensive subsystem component that will allow the capability of the integrated threads to operate in conjunction with each other as a single system component. To achieve this goal, each of the individual thread components, libraries, will be integrated together via a makefile and built by configuration management as one subsystem component. The following build testcases will demonstrate the integrity of the integration.

4.5.5.1 Test Case 1: User Registration Request (BC004.001)

This test case verifies the capability to submit a request by a guest user for a registered user account. Each request contains the information needed to create a user profile record. New user accounts are created and stored in the user profile database and new user accounts are created.

Test Configuration:

Hardware: MSS Server (msse4hp)

Software: MsAcUsrProfile, MsAcUsrReqMgr, MsAcGuestUserReg

Data: Sample user information, runserver.csh/runclient.csh drivers

Tools: DCE (rgy_edit), Sybase

Test Input:

Runserver.csh driver inserting user profile data into Sybase.

Test Output:

Sybase will return successful creation and insertion into database.

Success Criteria:

This test is successful when the DCE account is created and also when the user appears in the MsAcUserProfile table in Sybase.

Test Procedures:

Test Case ID: BC004.001	
Test Name: User Registration Request	
Test Steps:	Comments:
1. Login into msse4hp in the Release A mini-DAAC	
2. Set your terminal environment display enter: **setenv DISPLAY <IP address>**	This sets the environment display for the perspective machine/monitor where the user is logged.
3 From the command line on msse4hp, enter: **DCE_login cell_admin**	The cell_admin privilege is needed for DCE in order to create a keytab file for the executables.
4. Enter password for the cell_admin DCE account.	
5. Enter **rgy_edit <return>**	The rgy-edit is a cell_admin function that can only be initialized once the user is in DCE.
6. At the **rgy_edit** prompt, enter: >ktadd -p cell_admin -f MsAcUsrProfileServer. keytab	
7. Enter password for cell_admin and quit.	
8. Enter **kdestroy** while in the DCE mode.	The kdestroy command needs to be initiated before exiting DCE because it destroys a principal's login context and associated credentials that are created with each DCE session.
9. Enter exit to end the DCE session.	
10.From the msse4hp command line, enter: **cd /usr/testa/bin/ACC**	This directory path has the executables for User Registration.

11.From the msse4hp command line, enter: **runserver.csh**	Enables the Profile Server.
12.The following messages will appear: **Server is running, DB is ready** **Registered User server is ready**	
13.From the msse4hp command line, enter: **ps**	This displays the status for the MsAcServer. Please note the PID Id because it will be needed in step#16.
14.From the msse4hp command line, enter: **runclient.csh**	This executable will create, register, add user <username> into Sybase.
15.The runclient.csh driver will do the following: **Insert Request for <username>** **Retrieve Request List for <username>** **Create Registered User for <username>** **List Registered User for <username>** **List User Profile for for <username>** **List Daily, Weekly, Monthly Usage for <username>**	Gives User Profile information. C-MSS-75020
16.From the msse4hp command line, enter: **kill -9 <process #>**	This terminates the MsAcServer.

4.5.5.2 Test Case 2: Interactive File Transfer (BC004.002)

The purpose of the this test is to verify that the CSS File Access Service provides basic connection oriented operations for interactive file transfers using the file transfer protocol (FTP). FTP allows read access to all users. The ability to send and receive both ASCII and binary files interactively and non-interactively between hosts is verified. This test also verifies that existing files are not overwritten if files with same name are requested for transfer.

Test Configuration:

Hardware: SUN Solaris 2.4, HPUX 9.05, and SGI:IRIX 5.3

Software: CsftFTPRelA.cxx, CsFtFTPRelA.h, ftptest.C

Data: ASCII/Binary files representative oif the science products available in Release A

Tools: OODCE 1.0, RogueWave Tools.h++ 6.1.0 or version 5.17

Test Input:

Perform an ASCII or binary file transfer interactively to simulated local and remote DAAC. Verify file consistencies and that no files were overwritten. Perform a file transfer to SCF host (infrastructure) via ftp. Verify file transfers between the mini-DAAC and the infrastructure.

Test Output:

Successful connection to both local and remote DAAC hosts. Complete file transfers interactively from the source to the destination successfully without lost data. Matched file sizes for files at both the sending and receiving sites. Appropriate time/date stamps. No error messages should be displayed.

Success Criteria:

This test is successful if files can be sent and received without errors, time/stamps, valid filesize match, checksums are valid, and files are not overwritten.

Test Procedures:

Test Case ID: BC004.002	
Test Name: Interactive File Transfer	
Test Steps:	Comments:
1. Login to local DAAC host	
2. Type: **setenv DISPLAY 'host IP address':0.0** ex. setenv DISPLAY 155.157.123.25:0.0	Setting visualization for the user host
3. Set view in clearcase: cleartool setview 'username'	View is set for 'username'
4. Perform a ftp command to a local host. Enter: **ftp** hostname. Provide a user name and a password.	If an authorized user name and password are valid, then successful permission to enter the file transfer program is generated. Hence: the "open" command could also be used for ftp connections. C-CSS-60520, C-CSS-60600
5. Verify the file transfer efficiency statistics is "on" Type: **verbose;** type again to turn "on" if status says "off "	Default is "on" Provides authentication to file transfer efficiency statistics locally and remotely C-CSS-60920
6. Check the local status: Type: **status**	Identifies the local connection host, mode, verbose status, store and receive status, case status, number of transmission status, number of default file name mappings, printing status, proxy connection if any, and case status
7. To ensure that the user does not overwrite a file (s) locally or remotely, check the current settings: (step 6) Type **runique**: for files being received Type **sunique**: for files being sent	no overwritten files via ftp (sent or received) Default is "off "
8. Type **status**	Verify the status is "on" for both runique and sunique
9. Select the file(s) to transferred locally **get** filename (one) **mget** filename (multiple)	Default mode is ASCII File (s) are transferred to the current directory
10. View output listing: **ls -al**	A listing of new file(s) transferred C-CSS-60510
11. Create an error scenario: cannot capture a file (s)	
12. Change transfer mode to binary: Type: **binary**	Response: Type set to I C-CSS-60610
13. Repeat step 9	
14. Type: **ls -al**	verification of binary transferred file(s) C-CSS-60510

15. Set mode to default (ASCII) Type: ascii	Type set to A C-CSS-60610
16. Select the file(s) to be received locally **put** or **send** filename (one) **mput** filename (multiple)	File (s) from the user's current directory is sent to a local host
17. Type: **bye**	Exits ftp. Places user at its initial state
18. Type: **ls -al**	The initial state should show a received file in its current directory
19. Create an error scenario: c cannot retrieve a file (s)	
20. Re-enter ftp: Step 4	
21. Change transfer mode to binary: Type: **binary**	
22. Repeat step 16	Binary files are sent
	satisfies: C-CSS-60500
23. Type: **bye**	Exits ftp. Places user at its initial state
24. Type: **ls -al**	verification of binary transferred file(s)
25 Again, Re-enter ftp: Step 4	
26. Transfer mode is (ASCII)	Default: ASCII
	satisfies: C-CSS-60500
27. Initiate a close: **close**	The ftp session is closed.
28. 'Open' a session: **open hostname;** provide user login name and password	Independent ftp client process is started. (Interactive mode gives immediate responses)
29. Perform steps 5-14	Completed file transfer and verification
30. Type: **close**	closes a ftp client session
31. Type: **bye**	Exits ftp
32. Intermittedly verify file transfer type: Enter: type	Response: Using " "mode to transfer files
33. Repeat scenario twice more	

4.5.5.3 Test Case 3: Transfer Kerberos Encrypted Files (BC004.003)

The purpose of the this test is to verify that kerberos software is capable of sending and receiving files with kerberos encrypted data between local and remote terminals via kftp. Also, the correct privileges to access controls is verified.

Test Configuration:

Hardware: SUN Solaris 2.4, HPUX 9.05, and SGI:IRIX 5.3

Software: Kerberos COTs, CsFtFTPRelA

Data: ASCII/ text data files

Tools: Network analyzer: Sniffer; encryption tool

Test Input:

Through the Kerberos application, a test data file is encrypted and transmitted across the FDDI network. The network analyzer attempts to access the encrypted data file, reviewed its contents, and verify a transfer between the terminals in use. Transfer attempts are made via kftp.

Test Output:

The network analyzer accesses the encrypted data file and displays a summary of data file.

Success Criteria:

This test is successful if the files are transferred between the local and remote hosts via kftp, and valid users obtain the information contained in the data files. The transferred data must be encrypted.

Test Procedures:

Test Case ID: BC004.003	
Test Name: Transfer Kerberos Encrypted Files	
Test Steps:	Comments:
1. Login to local host	
2. Type: **setenv DISPLAY 'host IP address':0.0** ex. Setenv DISPLAY 155.157.123.25:0.0	Setting visualization for the user host
3. Setview in clearcase: cleartool setview 'username'	View is set for 'username'
4. Login to the kerberos COTS	
5. Locate any data file within kerberos and change format to encryption [done by the encryption tool]	
6. Begin data capture with the network analyzer.	
7. Initiate the cable tester and filtering	
On the workstation, perform a file transfer	
8. Perform a ftp command to a local host. Enter: **kftp** hostname. Provide a user name and a password.	If an authorized user name and password are valid, then successful permission to enter the file transfer program C-CSS-60900
9. Verify the file transfer efficiency statistics is "on" Type: **verbose;** type again to turn "on" if status says "off "	Default is "on" Provides authentication to file transfer efficiency statistics locally and remotely
10. Check the local status: Type: **status**	
11. To ensure that the user does not overwrite a file (s) locally or remotely, check the current settings: (step4) Type **runique**: for files being received Type **sunique**: for files being sent	no overwritten files via ftp (sent or received)
12.Type: **status**	Verify the status is "on" for both runique and sunique
13. Select the file(s) to transfer locally **get** filename (one) **mget** filename (multiple)	File (s) are transferred to the current directory

14. View output listing: **ls -al**	A listing of new file(s) transferred
15. Create an error scenario	
16. Type: **close**	closes ftp connection
17. Stop capturing data via the network analyzer: F10 option or ESC	
18. Using the network analyzer, ensure that all the files transferred via kftp are encrypted. Save to disk.	
19. Login to another host (valid user), and attempt to access the file	Only those with authorization will be able to access the file
20. Redo: Steps 4-17	
21. Repeat scenarios with invalid users	No authentication

4.5.5.4 Test Case 4: ACL Persistent Database Store (BC004.004)

The purpose of this test case is to demonstrate the capability of the CSS Security Service to save and retrieve the ACL database onto persistent storage.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h ACL Editor, ACL Entry Editor

Data: Supplied by user input.

Tools: Sybase 10.01, OODCE 1.0.3, RogueWave Tools.h++ 6.1, BuilderXcessory 3.5.1p

Test Input:

Supplied by user input

Test Outputs

The expected results of this test demonstrate that the in the event of the loss of the ACL database server, CSS Security provides recovery of the ACL database and maintains the original ACL database settings.

Success Criteria

This test will be deemed successful when, after the recovery of the ACL database the original ACL database values remain as originally set.

Test Procedures

Test Case ID: BC004.004	
Test Name: ACL Persistent Database Store	
Test Steps:	Comments
1) Enter: cd /u**sr/testa/IT/tools/CSS/SEC**	Change directory for server.
2) Start server. Enter: **server_AclDB &**	

3) Verify that server is listening.	
4) Invoke ACL Editor.	
5) Create a new ACL database using the ACL Editor.	
6) Create 5 new acl resource members using the ACL Entry Editor.	Save acls onto persistent storage.
7) Verify ACL database created with 5 acl members.	Verify acls from ACL Editor display
8) Enter: **Kill -9 <Process ID>**	Simulated server SHUTDOWN.
9) Delete server entry via cdsbrowser **</.:/cdsDirNam/cdsServerEntryName>**	
10) Restart server. Enter: **server_AclDB &**	Simulated server REBOOT.
11) Verify that server is listening.	
12) Invoke ACL Editor.	
13) Verify ACL database and 5 acl resource members persist.	Retrieve acls from persistent storage.
14) Enter: **Kill -9 <Process ID>**	
	C-CSS-21150

4.5.5.5 Test Case 5: CSS to MSS Login Notification (BC004.005)

The purpose of this test case is to demonstrate the capability of the CSS Security Service to notify the MSS Management Agent service upon a predetermined number of unsuccessful login attempts.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: Supplied by user input

Tools: OODCE 1.0.3, RogueWave Tools.h++ 6.1, BuilderXcessory 3.5.1p

Test Input:

Input to this test include the EcSeLogin_3 executable.

Test Outputs

The expected results of this test include notification to the MSS Management Agent upon a predetermined number of unsuccessful login attempts.

Success Criteria

This test will be deemed successful when a predetermined number of unsuccessful login attempts successfully notify the MSS Management Agent service.

Test Procedures

Test Case ID: BC004.005	
Test Name: CSS To MSS Login Notification	
Test Steps:	Comments
1) Enter: **cd /usr/testa/IT/tools/CSS/SEC**	change directory
2) Enter: **klist**	Verify credentials ticket not created
3) Enter: **EcSeLogin_3**	Display Usage instructions
4) Enter: **EcSeLogin_3 <username>**	Enter Valid Username
5) Enter Password: **<password>**	Enter Valid password
6) Enter: klist	Verify credentials ticket created
7) Enter: kdestroy	Destroy Ticket
8) Enter: **EcSeLogin_3 <username>**	Enter Valid Username
9) Enter Password: **<password>**	Enter Invalid Password; Repeat 2 times. Only 3 attempts are allowed.
10) Verify **Exit** from Password prompt.	
11) Enter: **klist**	Verify credentials ticket not created
12) Verify DCE login attempts notify MSS Agent.	C-CSS-21105
13) Enter: **EcSeLogin_3 <username>**	Enter Invalid Username.
14) Verify Invalid username rejected by DCE	ERROR: User ID Failure: Registry Object Not found (DCE/sec)
15) Enter: **EcSeLogin_3 <username> <password>**	Enter Valid username and Valid password on one line.
16) Enter: **klist**	Verify credentials ticket created
17) Enter: kdestroy	Destroy Ticket
18) Enter: **EcSeLogin_3 <username> <password>**	Enter Valid username and Invalid password on one line
19) Enter: **klist**	Verify credentials ticket created
20) Enter: **kdestroy**	Destroy Ticket
21) Verify DCE login attempts notify MSS Agent.	C-CSS-21105

4.5.5.6 Test Case 6: Permission Schema Definition (BC004.006)

The purpose of this test case is to demonstrate that the CSS Security Service provides an API to define the permission schema associated with a server or resource.

Test Configuration:

Hardware: SunOS 5.4, HP 9.05

Software: Schema Editor

Data: Schema set

Tools: Sybase 10.01, OODCE 1.0, RogueWave Tools.h++ 6.1

Test Input:

None

Test Outputs

The expected results of this test include demonstrating the ability to define additional permission schema to the schema provided by DCE.

Success Criteria

This test will be deemed when additional permissions are successfully defined to the existing schema provided.

Test Procedures

Build Test Case ID: BC004.006	
Test Name: Permission Schema Definition	
Test Steps:	Comments
1. Enter: cd /u**sr/testa/IT/tools/CSS/SEC**	Change directory for server.
2. Start server. Enter: **server_perm &**	
3. Verify that server is listening.	Verify default permission schema.
4. Enter: **pe**	List current permissions available.
5. Login to client.	
6. Perform DCE login. **"dce_login <username>"**	
7. Enter DCE **password**	
8. Enter: cd /u**sr/testa/IT/tools/CSS/SEC**	Change directory on client.
9. Invoke Schema editor.	Bringup Schema Editor screen.
10. Enter: <**ACL DB Name>**	
11. Enter: **<Application Name>**	
12. Enter: **A**	Permission Token string 'A'. Token character 'c' cannot be used.
13. Enter: **Rel-A I&T Token Test**	Type new permission usage definition.
14. Press: **Add**	Add new permission token.
15. Press: **Apply**	Apply the new permission to ACL database.
16. Verify list of permissions reflects additional 'A' permission added	See Schema Editor menu
17. Invoke ACL Entry Editor	Bringup ACL Entry editor screen
18. Verify new permission is displayed in the list of available ACLs	
19. Enter: <**ACL Name**>	Displayed in the Entry Key view
20. Select ACL resource to give permissions	
21. Select **Single** for Entry Type	
22. Select Entry Permissions: **c, x, r, w, A**	Select new permission with some standard permissions.
23. Press: **Add**	
24. Verify selected permissions are displayed from the Acl Entry Editor menu.	
25. Press: **Apply**	Apply the permissions to ACL database.
26. Bringup acledit	
27. Enter: **pe**	List current ACL schema
28. Verify new permission token displayed.	
29. From Acl Schema Editor screen, again bringup ACL DB Name:	ACL DB name from step 11
30. Select the '**w**' token (write permission)	

31. Press: **change**	
32. Change w to **B**	B is a user defined permission
33. Press: **Apply**	Apply the change to the ACL database
34. Verify the change from the ACL Entry Editor display.	See Entry Permission list
35. From acledit Enter: **pe**	List current ACL Schema
36. Verify permission **B** is displayed.	
37. From Acl Schema Editor screen, again bringup ACL DB Name:	ACL DB name from step 11
38. Select the '**B**' token.	B is a user defined permission
39. Press: **Delete**	
40. Press: **Apply**	
41. Verify B permission deleted from the ACL Entry Editor display.	
42. From acledit Enter: **pe**	
43. Verify permission **B** is not displayed.	
44. Enter: **Kill -9 <Process ID>**	
	C-CSS-21130

4.6 Application Support Services Tests

The Application Supports Services tests verify the following thread and build:

- Time Service Thread (TC016)

4.6.1 Time Service Thread (TC016)

The purpose of this thread is to demonstrate the functionality made available and supported by the Time Service API.

4.6.1.1 Test Case 1: Get Current Time (TC016.001)

The purpose of the Get Current Time test is to demonstrate the functionality made available and supported by the Time Services API. The test obtains and validates the current time in the following formats:

- ASCII string containing current GMT/UTC time.
- Current time in timespec_t format (seconds since 00:00:00 GMT, January 1, 1970).
- Current time in tm format (seconds, minutes, hours, day of month, month of year, year-1900, day of week, day of year, daylight savings indicator).
- Current system time and inaccuracy in the format of a binary timestamp.

<u>Test Configuration:</u>

Hardware: HP and SUN hosts

Software: EcTiTimeService, GetAscGmtTime, GetSecNanoTime, GetTimeValues, GetTime, GetRWTime

Data: None

Tools: CDS browser-COTS, timetest

Test Input:

Inputs to this test case include calls to the API from the test tool requesting time in each of the formats listed above.

Test Output:

Outputs to this test case include responses from the API containing the current time in the formats listed above.

Success Criteria:

This test is successful when the format and accuracy of the response for each API are verified.

Test Procedures:

Test Case ID: TC016.001	
Test Name: Get Current Time	
Test Steps:	Comments:
1. Log into DCE on an HP-UX or a SUN machine (e.g., csse3hp for testing on HP platform and msse6sun for testing on SUN platform) by typing: **dce_login username password**	Starting the Get Current Time test on an HP-UX or a SUN platform.
2. Execute the test program by typing: **timetest**	The current time will be used in the following member functions: GetAscGmtTime, GetSecNanotime, GetTimeValues, GetTime, GetRWTime. Also note that no delta value will be computed since the current time is being used.
3. Select option **5** from the menu for current time.	No entry is added to CDS since the current time is retrieved directly from the Time Service API.
4. Verify a message similar to the following appears indicating the GetRWTime function was properly executed: ***mm/dd/yy hours:min:secs*** Press **RETURN** to go to the next step.	This function obtains the current local time, applies a delta if applicable, and returns the computed time value as a RWTime object. The date and time should correspond to the host machine's current system time. (CSS-25030, 25110)
5. Verify a message similar to the following appears indicating the GetAscGmtTime function was properly executed: **Corresponding GMT Time: *year-mth-day-hr:min:sec.fffinacurracy*** Press **RETURN** to go to the next step.	This function obtains the current time, applies a delta if applicable, and returns an ASCII string that expresses GMT time. The date and time should correspond to the host machine's current system time minus its time zone offset. (CSS-25030,25050, 25110)

6. Verify a message similar to the following appears indicating the GetSecNanoTime function was properly executed: **Seconds: #** **Nanoseconds: #** **Inaccuracy in Seconds: #** **Inaccuracy in Nanoseconds: #** **TDF: #** **Corresponding GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function obtains the current time, applies a delta if applicable, and converts it to a timespec_t structure. It also converts the associated inaccuracy into a timespec_t structure. The time, its associated inaccuracy and the time differential are returned to the calling function. These values are printed by the test driver and then passed to the MkBinTime function which will convert these values into to a binary time stamp. This binary time stamp is then converted into an ASCII string representing GMT time via the BinToAscGmt function. This string is then printed and its value should correspond to the host machine's current system time minus its time zone offset (CSS-25030,25040,25050, 25110)
7. Verify a message similar to the following appears indicating the GetTimeValues function was properly executed: **Seconds: #** **Minutes: #** **Hours: #** **Day of Month: #** **Month of Year: #** **Year: #** **Day of Year: #** **Corresponding GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function obtains the current local time, applies a delta if applicable, and converts it to a tm structure. The tm structure is returned to the calling function. The test driver prints the values store in the tm structure and passes the tm structure to the MkGMTTime function. This function will convert these values into a binary time stamp. This binary time stamp is then converted into an ASCII string representing GMT time via the BinToAscGmt function. This string is then printed and its value should correspond to the host machine's current system time. (CSS-25020, 25030, 25040, 25050, 25060, 25110)
8. Verify a message similar to the following appears indicating the GetTime function was properly executed: **Corresponding GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function obtains the current time, applies a delta if applicable, and returns the computed time value as a utc structure. This binary time stamp is then converted into an ASCII string representing GMT time via the BinToAscGmt function. This string is then printed and its value should correspond to the host machine's current system time minus its time zone offset. (CSS-25020, 25030, 25050, 25110)
9. Type **n** when asked if you wish to continue.	Exiting the test driver.
10. Type **exit** to logout of DCE.	Remember when logging out of DCE, type kdestroy.

4.6.1.2 Test Case 2: Absolute Time (TC016.002)

The purpose of the Absolute Time test is to demonstrate the Time Service's ability to calculate and maintain a simulated time given an absolute time value. It will then make the following time conversions utilizing the calculated delta time where appropriate:

- Convert a binary time to a timespec_t format of GMT time.
- Convert a binary time to a tm format of GMT time.
- Convert a binary time to an ASCII format of GMT time.
- Convert a binary time to a tm format of local time.
- Convert a timespec_t format to a binary format.

- Convert a relative timespec_t format to a relative binary format.
- Compute the sum of two binary times.
- Compute the difference between two binary times.
- Obtain the local time zone label and offset from GMT, given the UTC.
- Convert a tm structure that expresses GMT or UTC to a binary timestamp.
- Compare two binary timestamps or two relative binary timestamps and return the relationship.
- Converts a character string to a binary timestamp.

Test Configuration:

Hardware: HP and Sun hosts

Software: EcTiTimeService, ApplyDelta, CalculateDelta, CvtDeltaToBinary GmTime, BinToAscGmt, MkBinTime, MkBinRelTime, AddTime, SubTime, GetLocalZone, MkGMTTime, CmpMidTime, GetLocalTime, CvtStrToBin

Data: None

Tools: CDS browser-COTS, timetest

Test Input:

Actions are performed to verify that the CSS Time Service provides APIs to compute time from current time plus/subtract the delta time parameter. Inputs to this test case include calls to the API from the test tool demonstrating the convert time functionality provided by the CSS Time Service.

Test Output:

Outputs to this test case include responses from the API calls to calculate the delta value and convert, manipulate and retrieve various formats of time stamps.

Success Criteria:

This test is successful when the format and accuracy of the convert time for each API are verified and the correct time values returned include the delta time offset.

Test Procedures:

Test Case ID: TC016.002	
Test Name: Absolute Time	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	**Starting the Absolute Time test on an HP-UX or a SUN platform.**

2. Create one session (open a window in an x-term) and log into DCE by typing : **dce_login *username password***	
3. Start up cdsbrowser by typing: **cdsbrowser &**	
4. Create a second session on that machine (for testing on an HP-UX platform) and also log into DCE by typing: **dce_login username password**	Note: To test on a **SUN platform**, log into DCE on a SUN machine (e.g., csse3sun) and execute the timetest program on that machine.
5. On the second session, execute the test program by typing : **timetest**	
6. Select option **1** from the menu to place an absolute time value in the cell's namespace.	It is possible to place an absolute value in the namespace and instantiate an EcTiTime object with a parameter providing the name of the cds entry containing the value. It is also possible to simply create an EcTiTime object that has the absolute value to be used as one of its parameters. An absolute time is a point on a time scale. When a user enters an absolute time, a delta value will be calculated by subtracting the current time with the absolute time. This delta value will then be applied in the GetAscGmtTime, GetSecNanoTime, GetTimeValues, GetTime functions.
7. When prompted enter the cds entry name to be used by the time service: ***i.e. /.:/RelA/TestTime***	
8. When prompted if this is a new entry into the cds enter: **yes**	If yes is entered, the test driver will create an entry for the value given in step 9, otherwise it is assumed an entry already exists for the name given in step 9.
9. When prompted, enter the absolute time in the following format: **CCYY-MM-DD-hh:mm:ss.fff** **(e.g., 1995-11-04-12:10:30.000)**	
10. To bring up the HP CDS Browser-Attribute List, double click on the appropriate cds entry.	The Attribute List is displayed.
11. From the Attribute List, double click the attribute name.	The absolute time value is displayed. Note, this value represents an absolute time in GMT format so if the current local time is to be entered its time zone offset must be included in this value.
12. Verify a message similar to the following appears in the test driver window indicating the GetRWTime function was properly executed: ***mm/dd/yy hours:min:secs*** Press **RETURN** to go to the next step.	This function obtains the current local time, applies a delta if applicable, and returns the computed time value as a RWTime object. The date and time should correspond to the value displayed in the attribute list plus whatever time has passed during execution of the test and the system's time zone offset. (CSS-25030, 25070, 25110)

13. Verify a message similar to the following appears indicating the GetAscGmtTime function was properly executed: **Corresponding GMT Time:** ***year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function obtains the current time, applies a delta if applicable, and returns an ASCII string that expresses GMT time. The date and time should correspond to the value displayed in the attribute list plus whatever time has passed during execution of the test. (CSS-25030, 25050, 25070, 25110)
14. Verify a message similar to the following appears indicating the GetSecNanoTime function was properly executed: **Seconds: #** **Nanoseconds: #** **Inaccuracy in Seconds: #** **Inaccuracy in Nanoseconds: #** **TDF: #** **Corresponding GMT Time:** ***year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function obtains the current time, applies a delta if applicable, and converts it to a timespec_t structure. It also converts the associated inaccuracy into a timespec_t structure. The time, its associated inaccuracy and the time differential are returned to the calling function. These values are printed by the test driver and then passed to the MkBinTime function which will convert these values into to a binary time stamp. This binary time stamp is then converted into an ASCII string representing GMT time via the BinToAscGmt function. This string is then printed and its value should correspond to the value displayed in the attribute list plus whatever time has passed during execution of the test. (CSS-25030, 25040, 25050, 25070, 25110)
15. Verify a message similar to the following appears indicating the GetTimeValues function was properly executed: **Seconds: #** **Minutes: #** **Hours: #** **Day of Month: #** **Month of Year: #** **Year: #** **Day of Year: #** **Corresponding GMT Time:** ***year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function obtains the current local time, applies a delta if applicable, and converts it to a tm structure. The tm structure is returned to the calling function. The test driver prints the values store in the tm structure and passes the tm structure to the MkGMTTime function. This function will convert these values into to a binary time stamp. This binary time stamp is then converted into an ASCII string representing GMT time via the BinToAscGmt function. This string is then printed and its value should correspond to the value displayed in the attribute list plus whatever time has passed during execution of the test and the system's time zone offset (CSS-25020, 25030, 25040, 25050, 25060, 25070, 25110)
16. Verify a message similar to the following appears indicating the GetTime function was properly executed: **Corresponding GMT Time:** ***year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function obtains the current time, applies a delta if applicable, and returns the computed time value as a utc structure. This binary time stamp is then converted into an ASCII string representing GMT time via the BinToAscGmt function. This string is then printed and its value should correspond to the value displayed in the attribute list plus whatever time has passed during execution of the test. (CSS-25020, 25030, 25050,25070, 25110)

17. Verify a message similar to the following appears indicating the GmTime function was properly executed: **Seconds: #** **Minutes: #** **Hours: #** **Day of Month: #** **Month of Year: #** **Year: #** **Day of Year: #** **Corresponding GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function excepts a utc structure and a tm structure as input. It then converts the utc into a tm value and returns the tm structure. The test driver passes the GmTime function the utc value computed in the previous step. The tm structure's values are displayed by the test driver. The utc binary time stamp is then converted into an ASCII string representing GMT time via the BinToAscGmt function. This string is then printed and its value should correspond to the value displayed in the previous step. (CSS-25020, 25030, 25040, 25050, 25070, 25110)
18. Verify a message similar to the following appears indicating the MkBinTime function was properly executed: **Seconds: #** **Nanoseconds: #** **Inaccuracy in Seconds: #** **Inaccuracy in Nanoseconds: #** **TDF: #** **Corresponding GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function converts an input time value in timespec_t format into an utc structure. The timespec_t values used in the test driver are the ones used in step 14. The timespec_t structure's values are displayed by the test driver along with the inaccuracy and the tdf. The computed utc binary time stamp is then converted into an ASCII string representing GMT time via the BinToAscGmt function. This string is then printed and its value should correspond to the value displayed in step 14. (CSS-25020, 25030, 25040, 25050, 25070, 25110)
19. Verify a message similar to the following appears indicating the MkBinRelTime function was properly executed: **Corresponding GMT Time: 1582-10-15-01:00:00.00.000I0.000** Press **RETURN** to go to the next step.	This function takes a reltimespec_t structure and its associated inaccuracy and converts it into a binary time stamp. The test driver uses reltimespec_t of 1hour and no inaccuracy. The computed utc binary time stamp is then converted into an ASCII string representing GMT time via the BinToAscGmt function. This string is then printed and its value should correspond to 1:00:00 on Oct. 15 1582.. (CSS-25020, 25030, 25040, 25050, 25110)
20. Verify a message similar to the following appears indicating the AddTime function was properly executed: **Current Asc GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** **Asc GMT Time to be subtracted: 1582-10-15-01:00:00.00.000I0.000** **Asc GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function takes two utc time values, adds them together and returns their sum. The test driver uses the utc value returned in the previous step along with a utc value it obtains from a call to the GetTime function. These two values are printed via the BinAscGmt function. The value returned from the AddTime function is subsequently printed. (CSS-25020, 25030, 25040, 25050,25060, 25090, 25110)

21. Verify a message similar to the following appears indicating the SubTime function was properly executed: **Current Asc GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** **Asc GMT Time to be added: 1582-10-15-01:00:00.00.000I0.000** **Asc GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function takes two utc time values, adds them together and returns their difference. The test driver uses the same arguments from the previous step as arguments to the SubTime function.. These two values are printed via the BinAscGmt function. The value returned from the AddTime function is subsequently printed. (CSS-25020, 25030, 25040, 25050, 25090, 25110)
22. Verify a message similar to the following appears indicating the GetLocalZone function was properly executed: **tzlocal - *local zone abbreviation*** **tzoffset - #** **tzdaylight - #** Press **RETURN** to go to the next step.	This function takes a binary time stamp and computes its associated local time zone, the offset for that time zone, and whether it is on standard time or daylight savings time. The test driver calls the GetTime function in order to obtain the current time in utc format and this value is passed to the GetLocalZone function. The host machine's local time zone will be displayed (i.e. EDT), along with its offset and the daylight savings flag (1 if daylight savings). (CSS-25020, 25030, 25050, 25060, 25100, 25110)
23. Verify a message similar to the following appears indicating the MkGMTTime function was properly executed: **Seconds: #** **Minutes: #** **Hours: #** **Day of Month: #** **Month of Year: #** **Year: #** **Day of Year: #** **Corresponding GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function converts a tm structure into a utc structure. The test driver uses the tm structure from step 15 as input to the MkGMTTime function. The tm structure's values are printed along with the ASCII string representation (obtained from BinToAscGmt) of the utc time stamp. The results displayed should correspond to those in step 15. (25030, 25040, 25050, 25070, 25110)
24. Verify a message similar to the following appears indicating the CmpMidTime function was properly executed: **Asc GMT Time utc1: *year-mth-day-hr:min:sec.fff\inacurracy*** **Asc GMT Time utc2: *year-mth-day-hr:min:sec.fff\inacurracy*** ***result of comparison*** Press **RETURN** to go to the next step.	This functions takes two binary time stamps and compares their values. The result of the comparison is returned as an enumerated type. The test driver uses the utcs obtained in steps 22 and 23 to do the comparison. (CSS-25050, 25080, 25110)

25. Verify a message similar to the following appears indicating the GetLocalTime function was properly executed: **current time is:** ***year-mth-day-hr:min:sec.fff\inacurracy*** **Seconds: #** **Minutes: #** **Hours: #** **Day of Month: #** **Month of Year: #** **Year: #** **Day of Year: #** Press **RETURN** to go to the next step.	This function takes a utc time structure and converts it into a tm structure representing the local time. The test driver obtains the current time in utc format using the GetTime function. The tm structure returned from the GetLocalTime function is displayed. (25030, 25040, 25050, 25070, 25110)
26. Enter **y** and press **RETURN** when asked whether to continue or not.	
27. Select option **3** from the menu in order to create a time object containing the absolute time in the constructor.	
28. When prompted, enter the absolute time in the following format: **CCYY-MM-DD-hh:mm:ss.fff** **(e.g., 1995-11-04-12:10:30.000)**	
29. Verify steps 12-25 for the new time object.	When verifying these steps the cds browser should not be referenced because the absolute time is not entered there. All references to the attribute in the cdsbrowser now refer to the entry placed in the constructor.
30. Enter **n** when asked whether to continue or not.	This will remove all entries added to the cds and exit.
31. Type **exit** to logout of DCE.	Remember when logging out of DCE, type kdestroy.

4.6.1.3 Test Case 3: Relative Time (TC016.003)

The purpose of the Relative Time test is to demonstrate the ability to obtain and validate a simulated time given a relative time offset. It will then make the following time conversions utilizing the delta time where appropriate:

- Convert a binary time to a timespec_t format of GMT time.
- Convert a binary time to a tm format of GMT time.
- Convert a binary time to an ASCII format of GMT time.
- Convert a binary time to a tm format of local time.
- Convert a timespec_t format to a binary format.
- Convert a relative timespec_t format to a relative binary format.
- Compute the sum of two binary times.
- Compute the difference between two binary times.
- Obtain the local time zone label and offset from GMT, given the UTC.

- Convert a tm structure that expresses GMT or UTC to a binary timestamp.
- Compare two binary timestamps or two relative binary timestamps and return the relationship.
- Converts a character string to a binary timestamp.

Test Configuration:

Hardware: HP and Sun hosts

Software: EcTiTimeService, ApplyDelta, CalculateDelta, CvtDeltaToBinary GmTime, BinToAscGmt, MkBinTime, MkBinRelTime, AddTime, SubTime, GetLocalZone, MkGMTTime, CmpMidTime, GetLocalTime, CvtStrToBin

Data: None

Tools: CDS browser-COTS, timetest

Test Input:

Actions are performed to verify that the CSS Time Service provides APIs to compute time from current time plus/subtract the delta time parameter. Inputs to this test case include calls to the API from the test tool demonstrating the convert time functionality provided by the CSS Time Service.

Test Output:

Outputs to this test case include responses from the API calls to calculate the delta value and convert, manipulate and retrieve various formats of time stamps.

Success Criteria:

This test is successful when the format and accuracy of the convert time for each API are verified and the correct time values returned include the delta time offset.

Test Procedures:

Test Case ID: TC016.003	
Test Name: Relative Time	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	**Starting the Relative Time test on an HP-UX or a SUN platform.**
2. Create one session (open a window in an x-term) and log into DCE by typing : **dce_login username password**	
3. Start up cdsbrowser by typing: **cdsbrowser &**	
4. Create a second session on that machine (for testing on HP-UX platform) and also log into DCE by typing: **dce_login username password**	Note: To test on a **SUN platform**, log into DCE on a SUN machine and execute the timetest program on that machine (e.g., csse3sun).

5. On the second session, execute the test program by typing : **timetest**	A delta time (or a relative time) is a discrete time interval that is usually added to or subtracted from the current time. This delta value will be applied to obtain the GetAscGmtTime, GetSecNanotime, GetTimeValues, GetTime. Note: This timetest program must be run twice (one for (+) delta and the other for (-) delta). * To test on a **SUN platform**, execute the timetest program on the SUN host (e.g., csse3sun).
6. Select option **2** from the menu to place a relative time (delta time) value in the cell's namespace.	It is possible to place a relative (delta) value in the namespace and instantiate an EcTiTime object with a parameter providing the name of the cdsentry containing the value. It is also possible to simply create an EcTiTime object that has the relative value to be used as one of its parameters. When a user enters a relative time (a delta time), this relative time will be added to or subtracted from the current time. The calculated value will then be applied to obtain the GetAscGmtTime, GetSecNanotime, GetTimeValues, GetTime.
7. When prompted enter the cds entry name to be used by the time service: ***i.e. /.:/RelA/TestTime***	
8. When prompted if this is a new entry into the cds enter: "**yes**"	If yes is entered, the test driver will create an entry for the value given in step 9, otherwise it is assumed an entry already exists for the name given in step 9.
9. When prompted, enter the delta time in the following format : **[+|-]DD-hh:mm:ss.fff** **(e.g., 21-08:30:25.000)** Press **RETURN** to go to the next step.	
10. To bring up the HP CDS Browser-Attribute List, double click on the appropriate cds entry.	The Attribute List is displayed.
11. From the Attribute List, double click the attribute name.	The relative time value is displayed.
12. Verify a message similar to the following appears in the test driver window indicating the GetRWTime function was properly executed: ***mm/dd/yy hours:min:secs*** Press **RETURN** to go to the next step.	This function obtains the current local time, applies a delta if applicable, and returns the computed time value as a RWTime object. The date and time should correspond to the current system time plus the relative time given in the cds. (CSS-25030, 25070)
13. Verify a message similar to the following appears indicating the GetAscGmtTime function was properly executed: **Corresponding GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function obtains the current time, applies a delta if applicable, and returns an ASCII string that expresses GMT time. The date and time should correspond to current system time, plus the value displayed in the attribute list, minus the local time zone offset. (CSS-25030, 25050, 25070, 25110)

14. Verify a message similar to the following appears indicating the GetSecNanoTime function was properly executed: **Seconds: #** **Nanoseconds: #** **Inaccuracy in Seconds: #** **Inaccuracy in Nanoseconds: #** **TDF: #** **Corresponding GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function obtains the current time, applies a delta if applicable, and converts it to a timespec_t structure. It also converts the associated inaccuracy into a timespec_t structure. The time, its associated inaccuracy and the time differential are returned to the calling function. These values are printed by the test driver and then passed to the MkBinTime function which will convert these values into to a binary time stamp. This binary time stamp is then converted into an ASCII string representing GMT time via the BinToAscGmt function. This string is then printed and the date and time should correspond to current system time, plus the value displayed in the attribute list, minus the local time zone offset. (CSS-25030, 25040, 25050, 25070, 25110)
15. Verify a message similar to the following appears indicating the GetTimeValues function was properly executed: **Seconds: #** **Minutes: #** **Hours: #** **Day of Month: #** **Month of Year: #** **Year: #** **Day of Year: #** **Corresponding GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function obtains the current local time, applies a delta if applicable, and converts it to a tm structure. The tm structure is returned to the calling function. The test driver prints the values store in the tm structure and passes the tm structure to the MkGMTTime function. This function will convert these values into to a binary time stamp. This binary time stamp is then converted into an ASCII string representing GMT time via the BinToAscGmt function. This string is then printed and the date and time should correspond to current system time, plus the value displayed in the attribute list. (CSS-25020, 25030, 25040, 25050, 25060, 25070, 25110)
16. Verify a message similar to the following appears indicating the GetTime function was properly executed: **Corresponding GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function obtains the current time, applies a delta if applicable, and returns the computed time value as a utc structure. This binary time stamp is then converted into an ASCII string representing GMT time via the BinToAscGmt function. This string is then printed and the date and time should correspond to current system time, plus the value displayed in the attribute list, minus the local time zone offset. (CSS-25020, 25030, 25050,25070, 25110)
17. Verify a message similar to the following appears indicating the GmTime function was properly executed: **Seconds: #** **Minutes: #** **Hours: #** **Day of Month: #** **Month of Year: #** **Year: #** **Day of Year: #** **Corresponding GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function excepts a utc structure and a tm structure as input. It then converts the utc into a tm value and returns the tm structure. The test driver passes the GmTime function the utc value computed in the previous step. The tm structure's values are displayed by the test driver. The utc binary time stamp is then converted into an ASCII string representing GMT time via the BinToAscGmt function. This string is then printed and its value should correspond to the value displayed in the previous step. (CSS-25020, 25030, 25040, 25050, 25070, 25110)

18. Verify a message similar to the following appears indicating the MkBinTime function was properly executed: **Seconds: #** **Nanoseconds: #** **Inaccuracy in Seconds: #** **Inaccuracy in Nanoseconds: #** **TDF: #** **Corresponding GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function converts an input time value in timespec_t format into an utc structure. The timespec_t values used in the test driver are the ones used in step 14. The timespec_t structure's values are displayed by the test driver along with the inaccuracy and the tdf. The computed utc binary time stamp is then converted into an ASCII string representing GMT time via the BinToAscGmt function. This string is then printed and its value should correspond to the value displayed in step 14. (CSS-25020, 25030, 25040, 25050, 25070, 25110)
19. Verify a message similar to the following appears indicating the MkBinRelTime function was properly executed: **Corresponding GMT Time: 1582-10-15-01:00:00.00.000I0.000** Press **RETURN** to go to the next step.	This function takes a reltimespec_t structure and its associated inaccuracy and converts it into a binary time stamp. The test driver uses reltimespec_t of 1hour and no inaccuracy. The computed utc binary time stamp is then converted into an ASCII string representing GMT time via the BinToAscGmt function. This string is then printed and its value should correspond to 1:00:00 on Oct. 15 1582.. (CSS-25020, 25030, 25040, 25050, 25110)
20. Verify a message similar to the following appears indicating the AddTime function was properly executed: **Current Asc GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** **Asc GMT Time to be added: 1582-10-15-01:00:00.00.000I0.000** **Asc GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function takes two utc time values, adds them together and returns their sum. The test driver uses the utc value returned in the previous step along with a utc value it obtains from a call to the GetTime function. These two values are printed via the BinAscGmt function. The value returned from the AddTime function is subsequently printed. (CSS-25020, 25030, 25040, 25050,25060, 25090, 25110)
21. Verify a message similar to the following appears indicating the SubTime function was properly executed: **Current Asc GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** **Asc GMT Time to be subracted: 1582-10-15-01:00:00.00.000I0.000** **Asc GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function takes two utc time values, adds them together and returns their difference. The test driver uses the same arguments from the previous step as arguments to the SubTime function.. These two values are printed via the BinAscGmt function. The value returned from the AddTime function is subsequently printed. (CSS-25020, 25030, 25040, 25050, 25090, 25110)
22. Verify a message similar to the following appears indicating the GetLocalZone function was properly executed: **tzlocal - *local zone abbreviation*** **tzoffset - #** **tzdaylight - #** Press **RETURN** to go to the next step.	This function takes a binary time stamp and computes its associated local time zone, the offset for that time zone, and whether it is on standard time or daylight savings time. The test driver calls the GetTime function in order to obtain the current time in utc format and this value is passed to the GetLocalZone function. The host machine's local time zone will be displayed (i.e. EDT), along with its offset and the daylight savings flag (1 if daylight savings). (CSS-25020, 25030, 25050, 25060, 25100, 25110)

23. Verify a message similar to the following appears indicating the MkGMTTime function was properly executed: **Seconds: #** **Minutes: #** **Hours: #** **Day of Month: #** **Month of Year: #** **Year: #** **Day of Year: #** **Corresponding GMT Time: *year-mth-day-hr:min:sec.fff\inacurracy*** Press **RETURN** to go to the next step.	This function converts a tm structure into a utc structure. The test driver uses the tm structure from step 15 as input to the MkGMTTime function. The tm structure's values are printed along with the ASCII string representation (obtained from BinToAscGmt) of the utc time stamp. The results displayed should correspond to those in step 15. (25030, 25040, 25050, 25070, 25110)
24. Verify a message similar to the following appears indicating the CmpMidTime function was properly executed: **Asc GMT Time utc1: *year-mth-day-hr:min:sec.fff\inacurracy*** **Asc GMT Time utc2: *year-mth-day-hr:min:sec.fff\inacurracy*** ***result of comparison*** Press **RETURN** to go to the next step.	This functions takes two binary time stamps and compares their values. The result of the comparison is returned as an enumerated type. The test driver uses the utcs obtained in steps 22 and 23 to do the comparison. (CSS-25050, 25080, 25110)
25. Verify a message similar to the following appears indicating the GetLocalTime function was properly executed: **current time is: *year-mth-day-hr:min:sec.fff\inacurracy*** **Seconds: #** **Minutes: #** **Hours: #** **Day of Month: #** **Month of Year: #** **Year: #** **Day of Year: #** Press **RETURN** to go to the next step.	This function takes a utc time structure and converts it into a tm structure representing the local time. The test driver obtains the current time in utc format using the GetTime function. The tm structure returned from the GetLocalTime function is displayed. (25030, 25040, 25050, 25070, 25110)
26. Enter **y** and press **RETURN** when asked whether to continue or not.	
27. Select option **4** from the menu in order to create a time object containing the absolute time in the constructor.	
28. When prompted, enter the delta time in the following format: **[+|-]DD-hh:mm:ss.fff** **(e.g., 21-08:30:25.000)**	

29. Verify steps 12-25 for the new time object.	When verifying these steps the cds browser should not be referenced because the relative time is not entered there. All references to the attribute in the cdsbrowser now refer to the entry placed in the constructor.
30. Enter **n** when asked whether to continue or not.	This will remove all entries added to the cds and exit.
31. Type **exit** to logout of DCE.	Remember when logging out of DCE, type kdestroy.

4.7 Application Management Service Tests

Application Management Service Tests consist of the following integrated threads and builds:

- Lifecycle Services Thread (TC017)
- Startup/Shutdown Thread (TC018)
- MIB/Extensible Agent - Application 2 Thread (TC019)
- Process Framework Thread (TC040)
- Application Management Services Build (BC006)

4.7.1 MSS Lifecycle Services Thread (TC017)

The objectives of MSS Lifecycle Services thread are:

- to verify the system's ability to manually (or automatically) startup or shutdown a subagent due to some anomaly.
- to verify the system's ability to manually (or automatically) startup or shutdown a running application, program, process (can only be shutdown) or COTS.

In order to meet the test objectives, the following functional capabilities are required:

- OODCE
- HpOpenView and GUI interface to M&O operator
- Extensible ECS management agent for ECS Host systems and applications
- MSS lifecycle related functionality's for M&O to manage server applications (e.g. data server application…). Associated functional capabilities for allowing shutdown lifecycle services .
- Management of Configuration, Performance, and Fault metrics
- CSS provided internal APIs for applications to be controlled by MSS (Process Frame Work)
- Proxy agents for ECS network devices and applications that cannot be managed via SNMP

4.7.1.1 Test Case 1: Application Startup/Shutdown (TC017.001)

This test verifies the ability of the MSS Management Agent Service to provide an extensible ECS management agent for ECS applications and to handle the startup and shutdown of an application. The SubAgent's ability to tell an application to shutdown gracefully or immediately, along with killing an application, will be verified. When the subagent kills an application, the application will receive no notification prior to its termination.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP), Inter-networking subsystem components (i.e. LAN Network resources)

Software: HpOpenView, OODCE, MACI: MsAgSubAgent, MsAgDeputy, MsAgAgent (Peer SNMP), CSS APIs (MPF)

Data: N/A

Tools: HpOpenView MIB browser (verify startup/shutdown in the tables), Functional applications: MsAgSubAgent, MsAgDeputy, MsAgAgent, Test drivers: dgtest (deputy gate controller), mpProgram (example program that has been developed with the Managed Process Framework library. The same executable can be used to simulate multiple applications. It contains the following capabilities: shutdown on request from the Agent, shutdown itself if start up from command line (through interactive interface), also send events if start up from command line (through interactive interface), register configuration, fault, and performance metrics. Once started, this program will read in the MSS Metric file, and connect with the subagent running on the local host (if present)).

Test Input:

Inputs to this test case include telling an application to startup, shutdown and killing an application.

Test Output:

Outputs include notification that an application start up, shutdown or is killed by streaming an event to the log file and by notification to HPOV.

Success Criteria

This test is successful when the application can startup, shutdown, and can be killed.

Test Procedures:

Test Case ID: TC017.001	
Test Name: Application Startup/Shutdown	
Test Steps:	Comments:
a. Start up PEER master agent, deputy, subagent, dgtest driver, and HPOV MIB Browser:	If they are not up.
1. Enter the PEER master agent directory: **cd /usr/testa/msagent2/bin**	
2. Start the master agent in the background: **sudo MsAgAgent ../data/MsAgAgent.cfg** *<filename>* **ctrl z** **bg**	MsAgAgent - PEER master agent (COTS Application) Note: This requires root or sudo root access. Any filename is acceptable.
3. Open a window on an HP machine where MsAgDeputy resides and log into DCE by typing : **dce_login** *<username>*	This application resides on the management host, and receives events from all subagents within each DAAC. The deputy converts all events to SNMP Traps, and sends them to HPOV's trapd.
4. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the Deputy.
5. Start up the MsAgDeputy by typing: **MsAgDeputy**	In order to start the Deputy, all required environment variables must be set (as described in the installation process).
6. Create another session (on SUN or HP platform) and also log into DCE by typing: **dce_login** *<username>*	
7. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the SubAgent.
8. Start up the MsAgSubAgent by typing: **MsAgSubAgent ConfigFile MsAgSubAgent.aCfg ecs_mode ops**	This application has the following capabilities: - Discovery of installed/DeInstalled/Upgraded applications and programs - Management of ECS developed applications and Proxy Agents - Event notification from application to subagent - Lifecycle services (startup, shutdown) - Metric handling (performance, fault, conf) - Event Notification to MsAgDeputy (HPOV) In order to start the SubAgent, all required environment variables must be set (as described in the installation process). Once started, the SubAgent will identify all installed Applications, Programs, and previously managed Process (which are still running). An event will be generated for each identified managed object. All the entries in the Static Application and Static Program tables will be displayed prior to the subagent listening for RPC request.
9. Create a third session and also log into DCE by typing: **dce_login** *<username>*	

10. Start up dgtest test driver by typing: **dgtest** (By default, this program will connect with a subagent running on the same host the dgtest was started from) or: **dgtest** *<hostname>* (This example will connect the dgtest program to a subagent running on the host).	dgtest - deputy gate controller (test driver) Since the agent has not yet been integrated into HPOV, this test driver simulates all agent control functionality that HPOV will possess in Release A. Functionality includes: - Start & stop of Applications and Programs - Get the number of seconds it will take to shutdown a Program, Process, or Application - Shut down Applications, Programs, Processes - Shut down the Subagent - Set attributes
11. Start the HpOpenView MIB browser (on the machine where MIB browser resides): **/usr/OV/bin/xnmbrowser**	
12. Enter the Host name in the box entitled "Name or IP Address".	Also, enter the IP Address instead of the Host name (i.e. 155.157.123.18 for msse4hp or 155.157.123.15 for msse8sun) in the box entitled "Name or IP Address".
b. Testing:	
Scenario I: - startup & shutdown 1 application.	
13. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises.** **ecs.mss.agent.application.appStTable.appStEntry**	
14. From the sub-branch, query appStIndex	Note: to get the index of the application to be started from the static application table.
15. From the main menu of dgtest, press 2 to startup.	
16. From the submenu of dgtest, select 1 to startup the application.	
17. Specify the index (from step 14) to start.	Check the event, startup the application, is logged to the mss.log and is notified to HPOV.
18. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises.** **ecs.mss.agent.application.appDyTable.appDy**	Make sure the application has been started successfully and to get the index of the application to be shutdown from the dynamic application table.
19. From the main menu of dgtest, press 3 to shutdown.	
20. From the submenu, press 1 to shutdown the application.	Note: Specify the index (from step 18) to shutdown. Check the event, shutdown the application, is logged to the mss.log and is notified to HPOV.
21. From the MIB browser, query the dynamic sub-branches (all tables, e.g., ApplDyn, ProgDyn, ProcessDyn, all metric tables, except static tables, e.g., progTable, AppStTable).	All references to Application, its programs and processes are gone.
22. Verify all applicable processes are gone by executing the following command: **ps -ef | grep** *<processids # >*(under the application)	(Verify ability to provide an extensible ECS management agent for ECS applications - MSS-36090)

Scenario II: - startup application 1 - startup application 2 - startup application 3 - startup application 4 - shutdown application 2 - shutdown application 3 - shutdown program 1 of application 1 - shutdown a process of application 1 - kill a process of application 4 - shutdown application 4	all 4 applications are up. All references to application 2 and 3, their programs and processes are gone. Program 1 and its process are gone. The process and program of application 1 is gone. The process and program of application 4 is gone. The remaining 2 programs and processes of application 4 are gone.
23. Repeat steps 13 through 22 for scenario II.	
c. Exit:	
24. To exit this test, check the process id number of the following running jobs: **ps -ef | grep MsAg**	
25. Then, kill the master agent (MsAgAgent), the deputy (MsAgDeputy), the subagent (MsAgSubAgent) processes by executing the following command: **sudo kill -9** *<processids #>*	Note: It requires root or sudo root access to kill the master agent.
26. Check and make sure that all processes have been killed by: **ps -ef | grep MsAg**	
27. Next, make sure to clean up the CDS with this entry. You can do this by following the directions below: - Log into the HP machine where CDS resides - Log into DCE (dce_login) - Start up cdsbrowser by typing: **cdsbrowser** - Select "/.:/subsys", then "/.:subsys/HP" - Select "/.:/subsys/HP/sample-apps" - Select the object names that were registered (MsAgSubAgent and MsAgDeputy). They are: /.:/subsys/HP/sampleapps/MsAgSubAgent /.:/subsys/HP/sample-apps/MsAgDeputy - Highlight the entry and press the right button of your mouse to get a menu. Select "delete entry" to remove the entry from CDS. - Select all applications, programs, and processes under ops and test mode that were registered and delete them all: /.:/ecs/GSFC/ops/xProgram_ops /.:/ecs/GSFC/ops/xProgram_ops_group /.:/ecs/GSFC/OPS/xProgram_ops_profile Remember when logging out of DCE, type kdestroy.	

4.7.1.2 Test Case 2: Program Startup/Shutdown (TC017.002)

This test verifies the ability of the MSS Management Agent Service to provide an extensible ECS management agent for ECS applications and to handle the startup and shutdown of a program. The SubAgent's ability to tell a program to shutdown gracefully or immediately, along

with killing a program, will be verified. When the subagent kills a program, the program will receive no notification prior to its termination.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP), Inter-networking subsystem components (i.e. LAN Network resources)

Software: HpOpenView, OODCE, MACI: MsAgSubAgent, MsAgDeputy, MsAgAgent (Peer SNMP), CSS APIs (MPF)

Data: N/A

Tools: HpOpenView MIB browser (verify startup/shutdown in the tables), Functional applications: MsAgSubAgent, MsAgDeputy, MsAgAgent, Test drivers: dgtest (deputy gate controller), mpProgram (example program that has been developed with the Managed Process Framework library. The same executable can be used to simulate multiple applications. It contains the following capabilities: shutdown on request from the Agent, shutdown itself if start up from command line (through interactive interface), also send events if start up from command line (through interactive interface), register configuration, fault, and performance metrics. Once started, this program will read in the MSS Metric file, and connect with the subagent running on the local host (if present)).

Test Input:

Inputs to this test case include telling a program to startup, shutdown, and killing a program.

Test Output:

Outputs include notification that a program starts up, shuts down or is killed by streaming an event to the log file and by notification to HPOV.

Success Criteria

This test is successful when the program can startup, shut down, and can be killed.

Test Procedures:

Test Case ID: TC017.002	
Test Name: Program Startup/Shutdown	
Test Steps:	Comments:
a. Start up PEER master agent, deputy, subagent, dgtest driver, and HPOV MIB Browser:	If they are not up.
1. Enter the PEER master agent directory: **cd /usr/testa/msagent2/bin**	

2. Start the master agent in the background: **sudo MsAgAgent ../data/MsAgAgent.cfg** *<filename>* **ctrl z** **bg**	MsAgAgent - PEER master agent (COTS Application) Note: This requires root or sudo root access. Any filename is acceptable.
3. Open a window on an HP machine where MsAgDeputy resides and log into DCE by typing : **dce_login** *<username>*	This application resides on the management host, and receives events from all subagents within each DAAC. The deputy converts all events to SNMP Traps, and sends them to HPOV's trapd.
4. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the Deputy.
5. Start up the MsAgDeputy by typing: **MsAgDeputy**	In order to start the Deputy, all required environment variables must be set (as described in the installation process).
6. Create another session (on SUN or HP platform) and also log into DCE by typing: *>dce_login <username>*	
7. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the SubAgent.
8. Start up the MsAgSubAgent by typing: **MsAgSubAgent ConfigFile MsAgSubAgent.aCfg ecs_mode ops**	This application has the following capabilities: - Discovery of installed/DeInstalled/Upgraded applications and programs - Management of ECS developed applications and Proxy Agents - Event notification from application to subagent - Lifecycle services (startup, shutdown) - Metric handling (performance, fault, conf) - Event Notification to MsAgDeputy (HPOV) In order to start the SubAgent, all required environment variables must be set (as described in the installation process). Once started, the SubAgent will identify all installed Applications, Programs, and previously managed Process (which are still running). An event will be generated for each identified managed object. All the entries in the Static Application and Static Program tables will be displayed prior to the subagent listening for RPC request.
9. Create a third session and also log into DCE by typing: **dce_login** *<username>*	

10. Start up dgtest test driver by typing: **dgtest** (By default, this program will connect with a subagent running on the same host the dgtest was started from) or: **dgtest** *<hostname>* (This example will connect the dgtest program to a subagent running on the host).	dgtest - deputy gate controller (test driver) Since the agent has not yet been integrated into HPOV, this test driver simulates all agent control functionality that HPOV will possess in Release A. Functionality includes: - Start & stop of Applications and Programs - Get the number of secs it will take to shutdown a Program, Process, or Application - Shut down Applications, Programs, Processes - Shut down the Subagent - Set attributes
11. Start the HpOpenView MIB browser (on an HP machine that the MIB browser resides): **/usr/OV/bin/xnmbrowser**	
12. Enter the Host name in the box entitled "Name or IP Address".	Also, enter the IP Address instead of the Host name (i.e. 155.157.123.18 for msse4hp or 155.157.123.15 for msse8sun) in the box entitled "Name or IP Address".
b. Testing:	
Scenario I: - Startup & shutdown 1 program.	
13. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises.ecs.mss.agent.program.progTable.progStEntry**	
14. From the sub-branch, query progStIndex	Note: to get the index of the program to be started from the static program table.
15. From the main menu of dgtest, press 2 to startup.	
16. From the submenu, select 2 to startup the program.	
17. Specify the index (obtained from step 14) to start.	Check the event, startup the program, is logged to the mss.log and is notified to HPOV.
18. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises.ecs.mss.agent.program.progDyTable.progDy**	Make sure the program has been started successfully and to get the index of the program to be shutdown from the dynamic program table.
19. From the main menu of dgtest, press 3 to shutdown.	
20. From the submenu, press 2 to shutdown the program.	Note: Specify the index to shutdown (from step 17). Check the event, shutdown the program, is logged to the mss.log and is notified to HPOV.
21. From the MIB browser, query the dynamic sub-branches (all tables, e.g., ProgDyn, ProcessDyn, all metric tables, except static tables, e.g., progTable).	All references to programs and processes are gone. Note: If all programs are gone, then all references to Application of that program should be removed from metric and dynamic tables.
22. Verify the applicable process is gone by executing the following command: **ps -ef | grep** *<processid#>*(under the program)	(Verify ability to provide an extensible ECS management agent for ECS applications - MSS-36090)

Scenario II: - startup program 1 - startup program 2 - startup program 5 - shutdown program 2 - shutdown the process of program 5	All three programs are up. Program 2 and its process are gone. Process of program 5 and program 5 are gone. Program 1 should still be there.
23. Repeat steps 13 through 22 for scenario II.	
c. Exit:	
24. To exit this test, check the process id number of the following running jobs: **ps -ef | grep MsAg**	
25. Then, kill the master agent (MsAgAgent), the deputy (MsAgDeputy), the subagent (MsAgSubAgent) processes by executing the following command: **sudo kill -9** *<processids #>*	Note: It requires root or sudo root access to kill the master agent.
26. Check and make sure that all processes have been killed by: **ps -ef | grep MsAg**	
27. Next, make sure to clean up the CDS with this entry. You can do this by following the directions below: - Log into the HP machine where CDS resides - Log into DCE (dce_login) - Start up cdsbrowser by typing: **cdsbrowser** - Select "/.:/subsys", then "/.:subsys/HP" - Select "/.:/subsys/HP/sample-apps" - Select the object names that were registered (MsAgSubAgent and MsAgDeputy). They are: /.:/subsys/HP/sampleapps/MsAgSubAgent /.:/subsys/HP/sample-apps/MsAgDeputy - Highlight the entry and press the right button of your mouse to get a menu. Select "delete entry" to remove the entry from CDS. - Select all applications, programs, and processes under ops and test mode that were registered and delete them all: /.:/ecs/GSFC/ops/xProgram_ops /.:/ecs/GSFC/ops/xProgram_ops_group /.:/ecs/GSFC/ops/xProgram_ops_profile Remember when logging out of DCE, type kdestroy.	

4.7.1.3 Test Case 3: Process Shutdown (TC017.003)

This test verifies the ability of the MSS Management Agent Service to provide an extensible ECS management agent for ECS applications and to handle the shutdown of a process. The SubAgent's ability to tell a process to shutdown gracefully or immediately, along with killing a process, will be verified. When the subagent kills a process, the process will receive no notification prior to its termination.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP), Inter-networking subsystem components (i.e. LAN Network resources)

Software: HpOpenView, OODCE, MACI: MsAgSubAgent, MsAgDeputy, MsAgAgent (Peer SNMP), CSS APIs (MPF)

Data: N/A

Tools: HpOpenView MIB browser (verify startup/shutdown in the tables), Functional applications: MsAgSubAgent, MsAgDeputy, MsAgAgent, Test drivers: dgtest (deputy gate controller), mpProgram (example program that has been developed with the Managed Process Framework library. The same executable can be used to simulate multiple applications. It contains the following capabilities: shutdown on request from the Agent, shutdown itself if start up from command line (through interactive interface), also send events if start up from command line (through interactive interface), register configuration, fault, and performance metrics. Once started, this program will read in the MSS Metric file, and connect with the subagent running on the local host (if present)).

Test Input:

Inputs to this test case include telling the process to shutdown and killing the process.

Test Output:

Outputs include notification that the process shuts down or is killed by streaming an event to the log file and by notification to HPOV.

Success Criteria

This test is successful when the process can shutdown and can be killed.

Test Procedures:

Test Case ID: TC017.003	
Test Name: Process Shutdown	
Test Steps:	Comments:
a. Start up PEER master agent, deputy, subagent, dgtest driver, and HPOV MIB Browser:	If they are not up.
1. Enter the PEER master agent directory: **cd /usr/testa/msagent2/bin**	
2. Start the master agent in the background: **sudo MsAgAgent ../data/MsAgAgent.cfg** *<filename>* **ctrl z** **bg**	MsAgAgent - PEER master agent (COTS Application) Note: This requires root or sudo root access. Any filename is acceptable.

3. Open a window on an HP machine where MsAgDeputy resides and log into DCE by typing : **dce_login** *<username>*	MsAgDeputy - MSS Deputy (Application) This application resides on the management host, and receives events from all subagents within each DAAC. The deputy converts all events to SNMP Traps, and sends them to HPOV's trapd.
4. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the Deputy.
5. Start up the MsAgDeputy by typing: **MsAgDeputy**	In order to start the Deputy, all required environment variables must be set (as described in the installation process).
6. Create another session (on a SUN or HP platform) and also log into DCE by typing: **dce_login** *<username>*	
7. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the SubAgent.
8. Start up the MsAgSubAgent by typing: **MsAgSubAgent ConfigFile MsAgSubAgent.aCfg ecs_mode ops**	This application has the following capabilities: - Discovery of installed/DeInstalled/Upgraded applications and programs - Management of ECS developed applications and Proxy Agents - Event notification from application to subagent - Lifecycle services (startup, shutdown) - Metric handling (performance, fault, conf) - Event Notification to MsAgDeputy (HPOV) In order to start the SubAgent, all required environment variables must be set (as described in the installation process). Once started, the SubAgent will identify all installed Applications, Programs, and previously managed Process (which are still running). An event will be generated for each identified managed object. All the entries in the Static Application and Static Program tables will be displayed prior to the subagent listening for RPC request.
9. Create a third session and also log into DCE by typing: **dce_login** *<username>*	
10. Start up dgtest test driver by typing: **dgtest** (By default, this program will connect with a subagent running on the same host the dgtest was started from). or **dgtest** *<hostname>* (This example will connect the dgtest program to a subagent running on the host).	dgtest - deputy gate controller (test driver) Since the agent has not yet been integrated into HPOV, this test driver simulates all agent control functionality that HPOV will possess in Release A. Functionality includes: - Start & stop of Applications and Programs - Get the number of secs it will take to shutdown a Program, Process, or Application - Shut down Applications, Programs, Processes - Shut down the Subagent - Set attributes
11. Start the HpOpenView MIB browser (on an HP machine that MIB browser resides): **/usr/OV/bin/xnmbrowser**	

12. Enter the Host name in the box entitled "Name or IP Address".	Also, enter the IP Address instead of the Host name (i.e. 155.157.123.18 for msse4hp or 155.157.123.15 for msse8sun) in the box entitled "Name or IP Address".
b. Testing:	
13. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises.** **ecs.mss.agent.program.progTable.progStEntry**	
14. From the sub-branch, query progStIndex	Note: to get the index of the program to be started from the static program table.
15. From the main menu of dgtest, press 2 to startup.	
16. From the submenu, select 2 to startup the program.	
17. Specify the index (obtained from step 14) to start.	
18. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises.** **ecs.mss.agent.process.procDyTable.procDy** and query the index of the process to be shutdown from the dynamic process table.	
19. From the main menu of dgtest, press 3 to shutdown.	
20. From the submenu, press 3 to shutdown the process.	Check the event, shutdown the process, is logged to the mss.log and is notified to HPOV.
21. From the MIB browser, query the dynamic sub-branches (all tables, e.g., ProcessDyn, all metric tables, except static tables).	All references to processes are gone. Note: If all processes are gone, then all references to Application and programs should be removed from metric and dynamic tables.
22. Verify all applicable processes are gone by executing the following command: **ps -ef | grep** *<processid #>*	(Verify ability to provide an extensible ECS management agent for ECS applications - MSS-36090)
c. Exit:	
23. To exit this test, check the process id number of the following running jobs: **ps -ef | grep MsAg**	
24. Then, kill the master agent (MsAgAgent), the deputy (MsAgDeputy), the subagent (MsAgSubAgent) processes by executing the following command: **sudo kill -9** *<processids #>*	Note: It requires root or sudo root access to kill the master agent.
25. Check and make sure that all processes have been killed by: **ps -ef | grep MsAg**	
26. Next, make sure to clean up the CDS with this entry. You can do this by following the directions below: - Log into the HP machine where CDS resides - Log into DCE (dce_login) - Start up cdsbrowser by typing: **cdsbrowser** - Select "/.:/subsys", then "/.:subsys/HP" - Select "/.:/subsys/HP/sample-apps" - Select the object names that were registered (MsAgSubAgent and MsAgDeputy). They are: /.:/subsys/HP/sampleapps/MsAgSubAgent /.:/subsys/HP/sample-apps/MsAgDeputy - Highlight the entry and press the right button of your	

mouse to get a menu. Select "delete entry" to remove the entry from CDS. - Select all applications, programs, and processes under ops and test mode that were registered and delete them all: /.:/ecs/GSFC/ops/xProgram_ops /.:/ecs/GSFC/ops/xProgram_ops_group /.:/ecs/GSFC/ops/xProgram_ops_profile Remember when logging out of DCE, type kdestroy.	

4.7.1.4 Test Case 4: COTS Startup/Shutdown (TC017.004)

This test verifies the ability of the MSS Management Agent Service to handle the startup and shutdown of a COTS application. The SubAgent's ability to tell a COTS application to shutdown gracefully or immediately, along with killing a COTS application, will be verified. When the subagent kills a COTS application, the application will receive no notification prior to its termination.

In order to startup or shutdown an application which is a non-SNMP manageable COTS product, this application must be wrapped with a manageable Proxy Agent.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP), Inter-networking subsystem components (i.e. LAN Network resources)

Software: HpOpenView, OODCE, CSS APIs (MPF), NCSA HTTPD server, MACI: EcAgCOTSManager, EcAgErrPattern, EcAgPatternVec, EcAgProxyComm, EcAgProxy

Data: HTTPDProxy.AppCfg, HTTPDProxy.EcAgCfg

Tools: HpOpenView MIB browser (verify startup/shutdown in the tables), WWW Browser, HTTPDProxy.cxx

Test Input:

Inputs to this test case include the startup of the HTTPD Proxy Agent, the NCSA HTTPD server, and an ECS subagent.

Test Output:

Outputs include notification that a COTS product is started up or shutdown by streaming an event to the log file and by notification to HPOV.

Success Criteria

This test is successful when a COTS product can be started and shutdown.

Test Procedures:

Test Case ID: TC017.004	
Test Name: COTS Startup/Shutdown	
Test Steps:	Comments:
1. Run HTTPD start script.	HTTPD Proxy Agent starts. NCSA HTTPD server starts. HTTPD COTS Manager Object Instantiated and mapped to the HTTPD server.
2. Select 'Graceful Shutdown' option from the main menu.	The proxy agent shutdowns the NCSA HTTPD server after the server successfully completes all running tasks and notifies the event manager (EcAgEvent) that the NCSA HTTPD is shutting down.
3. Run the HTTPD start script.	HTTPD Proxy Agent starts. The Proxy Agent identifies NCSA HTTPD server, creates a HTTPD COTS Manager Objects, and reconnects to it.
4. Select 'Immediate Shutdown' option from the main menu.	The proxy agent immediately shutdown the NCSA HTTPD server and notifies the event manager (EcAgEvent) that the NCSA HTTPD is shutting down.
5. Run HTTPD start script.	HTTPD Proxy Agent starts. NCSA HTTPD server starts. HTTPD COTS Manager Object Instantiated and mapped to the HTTPD server.
6. Select 'kill' option from the main menu.	The proxy agent kills the NCSA HTTPD server without notification to the event manager (EcAgEvent). (Verify ability to provide proxy agents for ECS network devices and applications that cannot be managed via SNMP - MSS-36100)

4.7.1.5 Test Case 5: Proxy Agent Parsing COTS Logs Files (TC017.005)

This test case verifies the functionality of the ProxyAgent class library. Through the use of this library, a proxy agent is developed to manage the NCSA HTTPD server. This proxy agent parses the error log file of the HTTPD server, and maps log entries to EcAgEvents.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP), Inter-networking subsystem components (i.e. LAN Network resources)

Software: OODCE, CSS APIs (MPF), NCSA HTTPD server (COTS), MACI: EcAgCOTSManager, EcAgErrPattern, EcAgPatternVec, EcAgProxyComm, EcAgProxy,

Data: HTTPDProxy.ErrorPatterns (maps HTTPD log entries to EcAgEvents), HTTPDProxy.AppCfg, HTTPDProxy.EcAgCfg

Tools: WWW Browser, HTTPDProxy.cxx

Test Input:

Inputs to this test case include the startup of the web browser and accessing a page on the NCSA server.

Test Output:

Output includes events generated by the proxy agent.

Success Criteria:

This test is successful when errors generated by the HTTPD server are identified, and events are mapped to these errors.

Test Procedures:

Test Case ID: TC017.005	
Test Name: Proxy Agent Parsing COTS Logs Files	
Test Steps:	Comments:
1. Execute the NCSA HTTPD start script.	HTTPD Proxy Agent starts. NCSA HTTPD server starts. HTTPD COTS Manager Object instantiated and mapped to the HTTPD server.
2. Start web browser and access a page on the NCSA server that doesn't exist.	"File Does Not Exist" event is generated by the proxy agent.
3. Access a page that requires a username/password, and type in an illegal username/password	"Authorization: Type Mismatch" event is generated by the proxy agent.
4. Execute the NCSA HTTPD stop script.	NCSA HTTPD server stop. HTTPD COTS Manager Object exits. Verify ability to provide proxy agents for ECS network devices and applications that cannot be managed via SNMP - MSS-36100

4.7.1.6 Test Case 6: Subagent Startup/Shutdown (TC017.006)

This test verifies the ability of the MSS Management Agent Service to provide an extensible ECS management agent for ECS applications and to handle the startup and shutdown of a subagent. HPOV is responsible to startup and shutdown the subagent. The following scenarios will be verified:

- scenario I: No applications and programs are running when first start up the subagent. The subagent is then shut down and restarted (normal behavior).
- scenario II: Some applications and programs are started by the subagent.
  - a) All applications and programs are still running when the subagent is shut down and restarted.
  - b) Some applications and programs die before the subagent is restarted.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP), Inter-networking subsystem components (i.e. LAN Network resources)

Software: HpOpenView, OODCE, MACI: MsAgSubAgent, MsAgDeputy, MsAgAgent (Peer SNMP), CSS APIs (MPF)

Data: N/A

Tools: HpOpenView MIB browser (verify startup/shutdown in the tables), Functional applications: MsAgSubAgent, MsAgDeputy, MsAgAgent, Test drivers: dgtest (deputy gate controller), mpProgram (example program that has been developed with the Managed Process Framework library. The same executable can be used to simulate multiple applications. It contains the following capabilities: shutdown on request from the Agent, shutdown itself if start up from command line (through interactive interface), also send events if start up from command line (through interactive interface), register configuration, fault, and performance metrics. Once started, this program will read in the MSS Metric file, and connect with the subagent running on the local host (if present)).

Test Input:

Inputs to this test case include the startup and shutdown of the SubAgent.

Test Output:

Outputs include notification that the subagent is started, shutdown, and reconnected to all applications and programs by streaming events to the log file and by notification to HPOV.

Success Criteria

This test is successful when a subagent can be started up, shutdown, restarted and reconnected to all applications and programs that it communicated with before it went down.

Test Procedures:

Test Case ID: TC017.006	
Test Name: Subagent Startup/Shutdown	
Test Steps:	Comments:
a. Start up PEER master agent, deputy, subagent, and dgtest driver:	If they are not up.
1. Enter the PEER master agent directory: **cd /usr/testa/msagent2/bin**	
2. Start the master agent in the background: **sudo MsAgAgent ../data/MsAgAgent.cfg** *<filename>* **ctrl z** **bg**	MsAgAgent - PEER master agent (COTS Application) Note: This requires root or sudo root access. Any filename is acceptable.

3. Open a window on an HP machine where MsAgDeputy resides and log into DCE by typing : **dce_login** *<username>*	MsAgDeputy - MSS Deputy (Application) This application resides on the management host, and receives events from all subagents within each DAAC. The deputy converts all events to SNMP Traps, and sends them to HPOV's trapd.
4. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the Deputy.
5. Start up the MsAgDeputy by typing: **MsAgDeputy**	In order to start the Deputy, all required environment variables must be set (as described in the installation process).
6. Create another session (on a SUN or HP platform) and also log into DCE by typing: **dce_login** *<username>*	
7. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the SubAgent.
8. Start up the MsAgSubAgent by typing: **MsAgSubAgent ConfigFile MsAgSubAgent.aCfg ecs_mode ops**	This application has the following capabilities: - Discovery of installed/DeInstalled/Upgraded applications and programs - Management of ECS developed applications and Proxy Agents - Event notification from application to subagent - Lifecycle services (startup, shutdown) - Metric handling (performance, fault, conf) - Event Notification to MsAgDeputy (HPOV) In order to start the SubAgent, all required environment variables must be set (as described in the installation process). Once started, the SubAgent will identify all installed Applications, Programs, and previously managed Process (which are still running). An event will be generated for each identified managed object. All the entries in the Static Application and Static Program tables will be displayed prior to the subagent listening for RPC request. Check the event, startup the SubAgent, is logged to the mss.log and is notified to HPOV.
9. Create a third session and also log into DCE by typing: **dce_login** *<username>*	
10. Start up dgtest test driver by typing: **dgtest** (By default, this program will connect with a subagent running on the same host the dgtest was started from) or: **dgtest** *<hostname>* (This example will connect the dgtest program to a subagent running on the host).	dgtest - deputy gate controller (test driver) Since the agent has not yet been integrated into HPOV, this test driver simulates all agent control functionality that HPOV will possess in Release A. Functionality includes: - Start & stop of Applications and Programs - Get the number of secs it will take to

	shutdown a Program, Process, or Application - Shut down Applications, Programs, Processes - Shut down the Subagent - Set attributes
b. Testing:	
Scenario I: No applications and programs are running when startup and shutdown the subagent.	Normal behavior
11. From the main menu of dgtest, press 7 to shutdown the subagent.	Check the event, shutdown the SubAgent, is logged to the mss.log and is notified to HPOV.
12. Verify the subagent is gone by executing the following command: **ps -ef | grep** *<processid #>* (of the subagent)	
13. Restart the subagent by typing: **MsAgSubAgent ConfigFile MsAgSubAgent.aCfg ecs_mode ops**	(Verify ability to provide an extensible ECS management agent for ECS applications - MSS-36090)
Scenario II: Applications & programs are started by the subagent.	
Scenario IIa: All applications and programs are still running when subagent goes down & restarts.	
Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises. ecs.mss.agent.application.appStTable.appStEntry**	Start up an application by executing steps 14 through 19.
15. From the sub-branch, query appStIndex.	Note: to get the index of the application to be started from the static application table.
16. From the main menu of dgtest, press 2 to startup.	
17. From the submenu of dgtest, select 1 to startup the application.	
18. Specify the index (from step 14) to start.	Check the event, startup an application, is logged to the mss.log and is notified to HPOV.
19. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises. ecs.mss.agent.application.appDyTable.appDy**	Make sure the application has been started successfully by checking the Application Dynamic table.
20. Repeat steps 14 through 19 for starting up additional applications.	
21. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises. ecs.mss.agent.program.progTable.progStEntry**	Start up a program by executing steps 21 through 26.
22. From the sub-branch, query progStIndex	Note: to get the index of the program to be started from the static program table.
23. From the main menu of dgtest, press 2 to startup.	
24. From the submenu, select 2 to startup the program.	
25. Specify the index (obtained from step 22) to start.	Check the event, startup the program, is logged to the mss.log and is notified to HPOV.
26. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises. ecs.mss.agent.program.progDyTable.progDy**	Make sure the program has been started successfully.
27. Repeat steps 21 through 26 for starting up additional programs.	

28. From the main menu of dgtest, press 7 to shutdown the subagent.	Check the event, shutdown the SubAgent, is logged to the mss.log and is notified to HPOV.
29. Verify the subagent is gone by executing the following command: **ps -ef | grep** *<processid #>* (of the subagent)	
30. Restart the subagent by executing the following command: **MsAgSubAgent ConfigFile MsAgSubAgent.aCfg ecs_mode ops**	The subagent should rebind to all running applications and programs which it communicates before. Check that events are logged to the mss.log (Verify ability to provide an extensible ECS management agent for ECS applications - MSS-36090)
Scenario IIb: Some applications and programs die before subagent restart.	
31. From the main menu of dgtest, press 7 to shutdown the subagent.	Check the event, shutdown the SubAgent, is logged to the mss.log and is notified to HPOV.
32. Verify the subagent is gone by executing the following command: **ps -ef | grep** *<processid #>* (of the subagent)	
33. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises. ecs.mss.agent.application.appDyTable.appDy**	To get the index of the application to be shutdown from the dynamic application table.
34. From the main menu of dgtest, press 3 to shutdown.	
35. From the submenu, press 1 to shutdown an application.	Note: Specify the index (from step 31) to shutdown. Check the event, shutdown the application, is logged to the mss.log and is notified to HPOV.
36. From the MIB browser, query the dynamic sub-branches (all tables, e.g., ApplDyn, ProgDyn, ProcessDyn, all metric tables, except static tables, e.g., progTable, AppStTable).	All references to the application, its programs and processes are gone.
37. Verify all applicable processes are gone by executing the following command: **ps -ef | grep** *<processids # >*(under the application)	
38. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises. ecs.mss.agent.program.progDyTable.progDy**	To get the index of the program to be shutdown from the dynamic program table.
39. From the main menu of dgtest, press 3 to shutdown.	
40. From the submenu, press 2 to shutdown the program.	Note: Specify the index to shutdown (from step 36). Check the event, shutdown the program, is logged to the mss.log and is notified to HPOV.
41. From the MIB browser, query the dynamic sub-branches (all tables, e.g., ProgDyn, ProcessDyn, all metric tables, except static tables, e.g., progTable).	All references to programs and processes are gone. Note: If all programs are gone, then all references to Application of that program should be removed from metric and dynamic tables.
42. Verify the applicable process is gone by executing the following command: **ps -ef | grep** *<processid#>*(under the program)	

43. Restart the subagent by typing: **MsAgSubAgent ConfigFile MsAgSubAgent.aCfg ecs_mode ops**	The subagent should rebind to the remain running applications and programs which it communicates before and sends events to mss.log and to HPOV for those that died. (Verify ability to provide an extensible ECS management agent for ECS applications - MSS-36090)
c. Exit:	
44. To exit this test, check the process id number of the following running jobs: **ps -ef | grep MsAg**	
45. Then, kill the master agent (MsAgAgent), the deputy (MsAgDeputy), the subagent (MsAgSubAgent) processes by executing the following command: **kill -9 <processids #>**	Note: It requires root or sudo root access to kill the master agent.
46. Check and make sure that all processes have been killed by: **ps -ef | grep MsAg**	
47. Next, make sure to clean up the CDS with this entry. You can do this by following the directions below: - Log into the HP machine where CDS resides - Log into DCE (dce_login) - Start up cdsbrowser by typing: **cdsbrowser** - Select "/.:/subsys", then "/.:subsys/HP" - Select "/.:/subsys/HP/sample-apps" - Select the object names that were registered (MsAgSubAgent and MsAgDeputy). They are: /.:/subsys/HP/sampleapps/MsAgSubAgent /.:/subsys/HP/sample-apps/MsAgDeputy - Highlight the entry and press the right button of your mouse to get a menu. Select "delete entry" to remove the entry from CDS. - Select all applications, programs, and processes under ops and test mode that were registered and delete them all: /.:/ecs/GSFC/ops/xProgram_ops /.:/ecs/GSFC/ops/xProgram_ops_group /.:/ecs/GSFC/ops/xProgram_ops_profile Remember when logging out of DCE, type kdestroy.	

4.7.1.7 Test Case 7: Multiple SubAgents Communicate with one Deputy (TC017.007)

This test verifies the ability of the MSS Management Agent Service to provide an extensible ECS management agent for ECS applications and to handle multiple subagents from different hosts communicating with one deputy.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP), Inter-networking subsystem components (i.e. LAN Network resources)

Software: HpOpenView, OODCE, MACI: MsAgSubAgent, MsAgDeputy, MsAgAgent (Peer SNMP), CSS APIs (MPF)

Data: N/A

Tools: HpOpenView MIB browser (verify startup/shutdown in the tables), Functional applications: MsAgSubAgent, MsAgDeputy, MsAgAgent, Test drivers: dgtest (deputy gate controller), mpProgram (example program that has been developed with the Managed Process Framework library. The same executable can be used to simulate multiple applications. It contains the following capabilities: shutdown on request from the Agent, shutdown itself if start up from command line (through interactive interface), also send events if start up from command line (through interactive interface), register configuration, fault, and performance metrics. Once started, this program will read in the MSS Metric file, and connect with the subagent running on the local host (if present)).

Test Input:

Inputs to this test case include the startup of multiple SubAgents and one deputy.

Test Output:

Outputs include notification that multiple subagents are started up and communicate with the deputy by streaming events to the log file and to the deputy which converts events to traps and sends to HPOV.

Success Criteria

This test is successful when multiple subagents can be started and can communicate with the deputy.

Test Procedures:

Test Case ID: TC017.007	
Test Name: Multiple subagents communicate with one deputy	
Test Steps:	Comments:
a. Start up PEER master agent, deputy, and HPOV:	If they are not up.
1. Enter the PEER master agent directory: **cd /usr/testa/msagent2/bin**	
2. Start the master agent in the background: **sudo MsAgAgent ../data/MsAgAgent.cfg** *<filename>* **ctrl z** **bg**	MsAgAgent - PEER master agent (COTS Application) Note: This requires root or sudo root access. Any filename is acceptable.

3. Open a window on an HP machine where MsAgDeputy resides and log into DCE by typing : **dce_login** *<username>*	This application resides on the management host, and receives events from all subagents within each DAAC. The deputy converts all events to SNMP Traps, and sends them to HPOV's trapd.
4. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the Deputy.
5. Start up the MsAgDeputy by typing: **MsAgDeputy**	In order to start the Deputy, all required environment variables must be set (as described in the installation process).
6. Start the HpOpenView (on the machine where HPOV resides) by typing: **/usr/OV/bin/ovw**	
b. Testing:	
7. Create multiple sessions (on SUN and HP platforms) and also log into DCE by typing: **dce_login** *<username>*	
8. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the SubAgent.
9. Start up the SubAgents on different hosts by typing: **MsAgSubAgent ConfigFile MsAgSubAgent.aCfg ecs_mode ops**	When SubAgents start up, events are sent to mss.log and to the deputy which then convert them to traps and sent to HPOV. (Verify ability to provide an extensible ECS management agent for ECS applications - MSS-36090)
c. Exit:	
10. To exit this test, check the process id number of the following running jobs: **ps -ef | grep MsAg**	
11. Then, kill the master agent (MsAgAgent), the deputy (MsAgDeputy), the subagents (MsAgSubAgent) processes by executing the following command: **kill -9** *<processids #>*	Note: It requires root or sudo root access to kill the master agent.
12. Check and make sure that all processes have been killed by: **ps -ef | grep MsAg**	
13. Next, make sure to clean up the CDS with this entry. You can do this by following the directions below: - Log into the HP machine where CDS resides - Log into DCE (dce_login) - Start up cdsbrowser by typing: **cdsbrowser** - Select "/.:/subsys", then "/.:subsys/HP" - Select "/.:/subsys/HP/sample-apps" - Select the object names that were registered (MsAgSubAgent and MsAgDeputy). They are: /.:/subsys/HP/sampleapps/MsAgSubAgent /.:/subsys/HP/sample-apps/MsAgDeputy - Highlight the entry and press the right button of your mouse to get a menu. Select "delete entry" to remove the entry from CDS. - Select all applications, programs, and processes under ops and test mode that were registered and delete them all:	

/.:/ecs/GSFC/ops/xProgram_ops /.:/ecs/GSFC/ops/xProgram_ops_group /.:/ecs/GSFC/ops/xProgram_ops_profile Remember when logging out of DCE, type kdestroy.	

4.7.1.8 Test Case 8: Error Handling for SubAgent Talking to Deputy (TC017.008)

This test verifies the ability of the MSS Management Agent Service to provide error handling when the subagent communicates with the deputy. The following scenarios will be verified:

- Scenario I: If the deputy goes down, the subagent keeps the events and sends them out to the deputy when it is up again.
- Scenario II: The deputy goes down and the subagent also goes down after receiving some events. When they both come up again, the subagent will send the previous events to the deputy.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP), Inter-networking subsystem components (i.e. LAN Network resources)

Software: HpOpenView, OODCE, MACI: MsAgSubAgent, MsAgDeputy, MsAgAgent (Peer SNMP), CSS APIs (MPF)

Data: N/A

Tools: HpOpenView MIB browser (verify startup/shutdown in the tables), Functional applications: MsAgSubAgent, MsAgDeputy, MsAgAgent, Test drivers: dgtest (deputy gate controller), mpProgram (example program that has been developed with the Managed Process Framework library. The same executable can be used to simulate multiple applications. It contains the following capabilities: shutdown on request from the Agent, shutdown itself if start up from command line (through interactive interface), also send events if start up from command line (through interactive interface), register configuration, fault, and performance metrics. Once started, this program will read in the MSS Metric file, and connect with the subagent running on the local host (if present)).

Test Input:

Inputs to this test case include the startup and shutdown of the Deputy and the SubAgent.

Test Output:

Outputs include streaming events to the log file and to the deputy which converts events to traps and sends to HPOV.

Success Criteria

This test is successful when events are sent to the deputy after it is up again.

Test Procedures:

Test Case ID: TC017.008	
Test Name: Error Handling for SubAgent Talking to Deputy	
Test Steps:	Comments:
a. Start up PEER master agent, deputy, subagent, dgtest, and HPOV:	If they are not up.
1. Enter the PEER master agent directory: **cd /usr/testa/msagent2/bin**	
2. Start the master agent in the background: **sudo MsAgAgent ../data/MsAgAgent.cfg** *<filename>* **ctrl z** **bg**	MsAgAgent - PEER master agent (COTS Application) Note: This requires root or sudo root access. Any filename is acceptable.
3. Open a window on an HP machine where MsAgDeputy resides (msse4hp) and log into DCE by typing : **dce_login <username>**	This application resides on the management host, and receives events from all subagents within each DAAC. The deputy converts all events to SNMP Traps, and sends them to HPOV's trapd.
4. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the Deputy.
5. Start up the MsAgDeputy by typing: **MsAgDeputy**	In order to start the Deputy, all required environment variables must be set (as described in the installation process).
6. Create another session (on SUN or HP platform) and also log into DCE by typing: **dce_login** *<username>*	
7. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the SubAgent.
8. Start up the MsAgSubAgent by typing: **MsAgSubAgent ConfigFile MsAgSubAgent.aCfg ecs_mode ops**	This application has the following capabilities: - Discovery of installed/DeInstalled/Upgraded applications and programs - Management of ECS developed applications and Proxy Agents - Event notification from application to subagent - Lifecycle services (startup, shutdown) - Metric handling (performance, fault, conf) - Event Notification to MsAgDeputy (HPOV) In order to start the SubAgent, all required environment variables must be set (as described in the installation process). Once started, the SubAgent will identify all installed Applications, Programs, and previously managed Process (which are still running). An event will be generated for each identified managed object. All the entries in the Static Application and Static Program tables will be displayed prior to the subagent listening for RPC request.

9. Create another session and also log into DCE by typing: **dce_login** *<username>*	
10. Start up dgtest test driver by typing: **dgtest** (By default, this program will connect with a subagent running on the same host the dgtest was started from). or **dgtest** *<hostname>* (This example will connect the dgtest program to a subagent running on the host).	dgtest - deputy gate controller (test driver) Since the agent has not yet been integrated into HPOV, this test driver simulates all agent control functionality that HPOV will possess in Release A. Functionality includes: - Start & stop of Applications and Programs - Get the number of secs it will take to shutdown a Program, Process, or Application - Shut down Applications, Programs, Processes - Shut down the Subagent - Set attributes
11. Start the HpOpenView (on the machine where HPOV resides): **/usr/OV/bin/ovw**	
b. Testing:	
Scenario I: Deputy goes down, subagent keeps events and sends them to deputy when it's up again.	
12. Kill the deputy by executing the following command: **kill -9** *<processid #>* (of the deputy)	
13. Go to the MIB branch entitled: .iso.org.dod.internet.private.enterprises. ecs.mss.agent.application.appStTable.appStEntry	To start an application.
14. From the sub-branch, query appStIndex.	Note: to get the index of the application to be started from the static application table.
15. From the main menu of dgtest, press 2 to startup.	
16. From the submenu of dgtest, select 1 to startup the application.	
17. Specify the index (from step 14) to start.	Events are sent to the SubAgent.
Restart the deputy by typing: **MsAgDeputy**	Expect all previous events are sent to the mss.log and to deputy after it's up again. (Verify ability to provide an extensible ECS management agent for ECS applications - MSS-36090)
Scenario II: Deputy is down, subagent also goes down after receiving some events. They both comes up and subagent sends previous events to deputy.	
18. Kill the deputy by executing the following command: **kill -9** *<processid #>* (of the deputy)	
19. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises.** **ecs.mss.agent.application.appStTable.appStEntry**	To start an application.
20. From the sub-branch, query appStIndex	Note: to get the index of the application to be started from the static application table.
21. From the main menu of dgtest, press 2 to startup.	
22. From the submenu of dgtest, select 1 to startup the application.	
23. Specify the index (from step 14) to start.	SubAgent should receives some events.

24. From the main menu of dgtest, press 7 to shutdown the subagent.	
25. Verify the subagent is gone by executing the following command: **ps -ef | grep** *<processid #>* (of the subagent)	
26. Restart the MsAgDeputy by typing: **MsAgDeputy**	
Restart the SubAgent by typing: **MsAgSubAgent ConfigFile MsAgSubAgent.aCfg ecs_mode ops**	SubAgent should send the previous events received to the mss.log and to the deputy.
27. Go to the HPOV to check for events sent to it.	(Verify ability to provide an extensible ECS management agent for ECS applications - MSS-36090)
c. Exit:	
28. To exit this test, check the process id number of the following running jobs: **ps -ef | grep MsAg**	
29. Then, kill the master agent (MsAgAgent), the deputy (MsAgDeputy), the subagent (MsAgSubAgent) processes by executing the following command: **sudo kill -9** *<processids #>*	Note: It requires root or sudo root access to kill the master agent.
30. Check and make sure that all processes have been killed by: **ps -ef | grep MsAg**	
31. Next, make sure to clean up the CDS with this entry. You can do this by following the directions below: - Log into the machine where CDS resides - Log into DCE (dce_login) - Start up cdsbrowser by typing: **cdsbrowser** - Select "/.:/subsys", then "/.:subsys/HP" - Select "/.:/subsys/HP/sample-apps" - Select the object names that were registered (MsAgSubAgent and MsAgDeputy). They are: /.:/subsys/HP/sampleapps/MsAgSubAgent /.:/subsys/HP/sample-apps/MsAgDeputy - Highlight the entry and press the right button of your mouse to get a menu. Select "delete entry" to remove the entry from CDS. - Select all applications, programs, and processes under ops and test mode that were registered and delete them all: /.:/ecs/GSFC/ops/xProgram_ops /.:/ecs/GSFC/ops/xProgram_ops_group /.:/ecs/GSFC/ops/xProgram_ops_profile Remember when logging out of DCE, type kdestroy.	

4.7.1.9 Test Case 9: Error Handling for EcAgManager Talking to SubAgent (TC017.009)

This test verifies the ability of the MSS Management Agent Service to provide error handling when the EcAgManager communicates with the SubAgent. The following scenarios will be verified:

Scenario 1: When the SubAgent goes down, the EcAgManager can't send events to the SubAgent. It keeps these events and sends them to the SubAgent when it comes up again.

Scenario 2: When both go down and come up again, EcAgManager send the previous events to the SubAgent.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP), Inter-networking subsystem components (i.e. LAN Network resources)

Software: HpOpenView, OODCE, MACI: MsAgSubAgent, MsAgDeputy, MsAgAgent (Peer SNMP), CSS APIs (MPF)

Data: N/A

Tools: HpOpenView MIB browser (verify startup/shutdown in the tables), Functional applications: MsAgSubAgent, MsAgDeputy, MsAgAgent, Test drivers: dgtest (deputy gate controller), mpProgram (example program that has been developed with the Managed Process Framework library. The same executable can be used to simulate multiple applications. It contains the following capabilities: shutdown on request from the Agent, shutdown itself if start up from command line (through interactive interface), also send events if start up from command line (through interactive interface), register configuration, fault, and performance metrics. Once started, this program will read in the MSS Metric file, and connect with the subagent running on the local host (if present)).

Test Input:

Inputs to this test case include the startup and shutdown of the SubAgent and the EcAgManager.

Test Output:

Outputs include streaming events to the log file and to the deputy which converts events to traps and sends to HPOV.

Success Criteria

This test is successful when events are sent to the deputy after it is up again.

Test Procedures:

Test Case ID: TC017.009	
Test Name: Error Handling for EcAgManager Talking to SubAgent	
Test Steps:	Comments:
a. Start up PEER master agent, deputy, dgtest, and HPOV:	If they are not up.

1. Enter the PEER master agent directory: **cd /usr/testa/msagent2/bin**	
2. Start the master agent in the background: **sudo MsAgAgent ../data/MsAgAgent.cfg** *<filename>* **ctrl z** **bg**	MsAgAgent - PEER master agent (COTS Application) Note: This requires root or sudo root access. Any filename is acceptable.
3. Open a window on an HP machine where MsAgDeputy resides and log into DCE by typing : **dce_login** *<username>*	This application resides on the management host, and receives events from all subagents within each DAAC. The deputy converts all events to SNMP Traps, and sends them to HPOV's trapd.
4. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the Deputy.
5. Start up the MsAgDeputy by typing: **MsAgDeputy**	In order to start the Deputy, all required environment variables must be set (as described in the installation process).
6. Create another session (on SUN or HP platform) and also log into DCE by typing: **dce_login** *<username>*	
7. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the SubAgent.
8. Create another session and also log into DCE by typing: **dce_login** *<username>*	
9. Start up dgtest test driver by typing: **dgtest** (By default, this program will connect with a subagent running on the same host the dgtest was started from). or **dgtest** *<hostname>* (This example will connect the dgtest program to a subagent running on the host).	dgtest - deputy gate controller (test driver) Since the agent has not yet been integrated into HPOV, this test driver simulates all agent control functionality that HPOV will possess in Release A. Functionality includes: - Start & stop of Applications and Programs - Get the number of seconds it will take to shutdown a Program, Process, or Application - Shut down Applications, Programs, Processes - Shut down the Subagent - Set attributes
10. Start the HpOpenView (on the machine where HPOV resides): **/usr/OV/bin/ovw**	
b. Testing:	
Scenario I: SubAgent goes down, EcAgManager waits until SubAgent comes up and sends events to it.	
11. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises.** **ecs.mss.agent.application.appStTable.appStEntry**	To start an application.
12. From the sub-branch, query appStIndex.	Note: to get the index of the application to be started from the static application table.
13. From the main menu of dgtest, press 2 to startup.	
14. From the submenu of dgtest, select 1 to startup the application.	
15. Specify the index (from step 12) to start.	

16. Start up the SubAgent by typing: **MsAgSubAgent ConfigFile MsAgSubAgent.aCfg ecs_mode ops**	EcAgManager should send the event (startup an application) to the SubAgent which then sends to mss.log.
17. Look at HPOV to make sure that the event is sent to it.	(Verify ability to provide an extensible ECS management agent for ECS applications - MSS-36090)
Scenario II: When both go down and come up again, EcAgManager send the previous events to the SubAgent.	
c. Exit:	
18. To exit this test, check the process id number of the following running jobs: **ps -ef | grep MsAg**	
19. Then, kill the master agent (MsAgAgent), the deputy (MsAgDeputy), the subagent (MsAgSubAgent) processes by executing the following command: **sudo kill -9** *<processids #>*	Note: It requires root or sudo root access to kill the master agent.
20. Check and make sure that all processes have been killed by: **ps -ef | grep MsAg**	
21. Next, make sure to clean up the CDS with this entry. You can do this by following the directions below: - Log into the HP machine where CDS resides - Log into DCE (dce_login) - Start up cdsbrowser by typing: **cdsbrowser** - Select "/.:/subsys", then "/.:subsys/HP" - Select "/.:/subsys/HP/sample-apps" - Select the object names that were registered (MsAgSubAgent and MsAgDeputy). They are: /.:/subsys/HP/sampleapps/MsAgSubAgent /.:/subsys/HP/sample-apps/MsAgDeputy - Highlight the entry and press the right button of your mouse to get a menu. Select "delete entry" to remove the entry from CDS. - Select all applications, programs, and processes under ops and test mode that were registered and delete them all: /.:/ecs/GSFC/ops/xProgram_ops /.:/ecs/GSFC/ops/xProgram_ops_group /.:/ecs/GSFC/ops/xProgram_ops_profile Remember when logging out of DCE, type kdestroy.	

4.7.1.10 Test Case 10: Error Handling for SubAgent & Deputy Startup (TC017.010)

This test verifies the ability of the MSS Management Agent Service to provide error handling for SubAgent and Deputy startup when configuration files do not exist or invalid mode is entered or configuration files contain errors.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP), Inter-networking subsystem components (i.e. LAN Network resources)

Software: HpOpenView, OODCE, MACI (MsAgSubAgent, MsAgDeputy, MsAgAgent (Peer SNMP), CSS APIs (MPF)

Data: N/A

Tools: HpOpenView MIB browser (verify startup/shutdown in the tables), Functional applications: MsAgSubAgent, MsAgDeputy, MsAgAgent, Test drivers: dgtest (deputy gate controller), mpProgram (example program that has been developed with the Managed Process Framework library. The same executable can be used to simulate multiple applications. It contains the following capabilities: shutdown on request from the Agent, shutdown itself if start up from command line (through interactive interface), also send events if start up from command line (through interactive interface), register configuration, fault, and performance metrics. Once started, this program will read in the MSS Metric file, and connect with the subagent running on the local host (if present)).

Test Input:

Inputs to this test case include non-existing configuration files, invalid mode, and configuration files that contain errors.

Test Output:

Outputs include graceful handling.

Success Criteria

This test is successful when no core dump have been found during startup the SubAgent and Deputy.

Test Procedures:

Test Case ID: TC017.010	
Test Name:	
Test Steps: Error Handling for SubAgent & Deputy Startup	Comments:
a. Start up PEER master agent:	If they are not up.
1. Enter the PEER master agent directory: **cd /usr/testa/msagent2/bin**	
2. Start the master agent in the background: **sudo MsAgAgent ../data/MsAgAgent.cfg** *<filename>* **ctrl z** **bg**	MsAgAgent - PEER master agent (COTS Application) Note: This requires root or sudo root access. Any filename is acceptable.
b. Testing:	

Scenario I: Non-existing configuration files or invalid mode	
3. Remove MsAgSubAgent.aCfg or MsAgDeputy.aCfg file from the /usr/testa/msagent2/data directory or enter an invalid mode.	
4. Open a window on an HP machine where MsAgDeputy resides and log into DCE by typing : **dce_login** *<username>*	This application resides on the management host, and receives events from all subagents within each DAAC. The deputy converts all events to SNMP Traps, and sends them to HPOV's trapd.
5. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the Deputy.
6. Start up the MsAgDeputy by typing: **MsAgDeputy**	If the MsAgDeputy.aCfg doesn't exist, the Deputy is not up and no core dump has been found. It should handle gracefully.
7. Create another session (on SUN or HP platform) and also log into DCE by typing: **dce_login** *<username>*	
8. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the SubAgent.
9. Start up the MsAgSubAgent by typing: **MsAgSubAgent ConfigFile MsAgSubAgent.aCfg ecs_mode ops**	If the MsAgSubAgent.aCfg doesn't exist or if the ecs_mode is invalid, the SubAgent is not up and no core dump has been found. It should handle gracefully.
Scenario II: Configuration files contain errors	
10. Open a window on an HP machine where MsAgDeputy resides and log into DCE by typing : **dce_login** *<username>*	This application resides on the management host, and receives events from all subagents within each DAAC. The deputy converts all events to SNMP Traps, and sends them to HPOV's trapd.
11. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the Deputy.
12. Start up the MsAgDeputy by typing: **MsAgDeputy**	If the MsAgDeputy.aCfg contains errors, the Deputy is not up and no core dump has been found. It should handle gracefully.
13. Create another session (on SUN or HP platform) and also log into DCE by typing: **dce_login** *<username>*	
14. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the SubAgent.
15. Start up the MsAgSubAgent by typing: **MsAgSubAgent ConfigFile MsAgSubAgent.aCfg ecs_mode ops**	If the MsAgSubAgent.aCfg contains errors, the SubAgent is not up and no core dump has been found. It should handle gracefully.
c. Exit:	
16. To exit this test, check the process id number of the following running jobs: **ps -ef | grep MsAg**	
17. Then, kill the master agent (MsAgAgent), the deputy (MsAgDeputy), the subagent (MsAgSubAgent) processes by executing the following command: **sudo kill -9** *<processids #>*	Note: It requires root or sudo root access to kill the master agent.
18. Check and make sure that all processes have been killed by: **ps -ef | grep MsAg**	

19. Next, make sure to clean up the CDS with this entry. You can do this by following the directions below: - Log into the HP machine where CDS resides - Log into DCE (dce_login) - Start up cdsbrowser by typing: **cdsbrowser** - Select "/.:/subsys", then "/.:subsys/HP" - Select "/.:/subsys/HP/sample-apps" - Select the object names that were registered (MsAgSubAgent and MsAgDeputy). They are: /.:/subsys/HP/sampleapps/MsAgSubAgent /.:/subsys/HP/sample-apps/MsAgDeputy - Highlight the entry and press the right button of your mouse to get a menu. Select "delete entry" to remove the entry from CDS. - Select all applications, programs, and processes under ops and test mode that were registered and delete them all: /.:/ecs/GSFC/ops/xProgram_ops /.:/ecs/GSFC/ops/xProgram_ops_group /.:/ecs/GSFC/ops/xProgram_ops_profile Remember when logging out of DCE, type kdestroy.	

4.7.2 Discover/Monitor Thread (TC018)

This test thread verifies the functionality of the MSS Discovery and Management Agent services. The purpose of this thread is to verify that this service is able to:

- discover new instances of managed objects.
- report new instances of managed objects.
- discover missing occurrences of managed objects.
- report missing occurences of managed objects.
- update the object database after the Discovery Service receives a request to register/unregister a managed object.

The MSS Management Agent Service is able to send ECS management traps/events and log events. The subagent can also instantiate the object responsible for local polling of resources on the host.

4.7.2.1 Test Case 1: MSS Discovery (TC018.001)

The purpose of this test case is to verify:

- the ability of the MSS Discovery Service to discover, via network protocol, new instances of managed objects, to detect and report missing occurrences of managed objects, and to update the object database (tables on the Agent side).
- the ability to retain the status of managed objects.

The discovery finds new instances of objects. It then creates an event and logs the event which MsAgDeputy converts to a trap and sends it to HPOV's trapd.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: HpOpenView, OODCE, MACI: MsAgSubAgent, MsAgDeputy, MsAgAgent (Peer SNMP), CSS APIs (MPF)

Data: N/A

Tools: HPOV

Functional applications: MsAgSubAgent, MsAgDeputy, MsAgAgent, Test drivers: dgtest (deputy gate controller), mpProgram (example program that has been developed with the Managed Process Framework library. The same executable can be used to simulate multiple applications. It contains the following capabilities: shutdown on request from the Agent, shutdown itself if start up from command line (through interactive interface), also send events if start up from command line (through interactive interface), register configuration, fault, and performance metrics. Once started, this program will read in the MSS Metric file, and connect with the subagent running on the local host (if present)).

Test Input:

N/A

Test Output:

Output includes subagent tables reflected change.

Success Criteria:

This test is successful when configured components and devices are monitored appropriately.

Test Procedures:

Test Case ID: TC018.001	
Test Name: MSS Discovery	
Test Steps:	Comments:
a. Start up PEER master agent, deputy, subagent, dgtest driver, and HPOV MIB Browser:	If they are not up.
1. Enter the PEER master agent directory: **cd /usr/testa/msagent2/bin**	

2. Start the master agent in the background: **sudo MsAgAgent ../data/MsAgAgent.cfg** *<filename>* **ctrl z** **bg**	MsAgAgent - PEER master agent (COTS Application) Note: This requires root or sudo root access. Any filename is acceptable.
3. Open a window on an HP machine where MsAgDeputy resides and log into DCE by typing : **dce_login** *<username>*	MsAgDeputy - MSS Deputy (Application) This application resides on the management host, and receives events from all subagents within each DAAC. The deputy converts all events to SNMP Traps, and sends them to HPOV's trapd.
4. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the Deputy.
5. Start up the MsAgDeputy by typing: **MsAgDeputy**	In order to start the Deputy, all required environment variables must be set (as described in the installation process).
6. Create another session (on SUN or HP platform) and also log into DCE by typing: **dce_login** *<username>*	
7. Before starting up the subagent, make sure the DISCOVER_INTERVAL in /usr/testa/msagent2/data/environment.csh file is changed to a small value (e.g., 120 or 180 secs).	
8. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the SubAgent.
9. Start up the MsAgSubAgent by typing: **MsAgSubAgent ConfigFile MsAgSubAgent.aCfg ecs_mode ops**	This application has the following capabilities: - Discovery of installed/DeInstalled/Upgraded applications and programs - Management of ECS developed applications and Proxy Agents - Event notification from application to subagent - Lifecycle services (startup, shutdown) - Metric handling (performance, fault, conf) - Event Notification to MsAgDeputy (HPOV) In order to start the SubAgent, all required environment variables must be set (as described in the installation process). Once started, the SubAgent will identify all installed Applications, Programs, and previously managed Process (which are still running). An event will be generated for each identified managed object. All the entries in the Static Application and Static Program tables will be displayed prior to the subagent listening for RPC request.
10. Create a third session and also log into DCE by typing: **dce_login** *<username>*	

11. Start up dgtest test driver by typing: **dgtest** (By default, this program will connect with a subagent running on the same host the dgtest was started from) or: **dgtest** *<hostname>* (This example will connect the dgtest program to a subagent running on the host).	dgtest - deputy gate controller (test driver) Since the agent has not yet been integrated into HPOV, this test driver simulates all agent control functionality that HPOV will possess in Release A. Functionality includes: - Start & stop of Applications and Programs - Get the number of secs it will take to shutdown a Program, Process, or Application - Shut down Applications, Programs, Processes - Shut down the Subagent - Set attributes
12. Start the HpOpenView MIB browser (on the machine where MIB browser resides): **/usr/OV/bin/xnmbrowser**	
13. Enter the Host name in the box entitled "Name or IP Address".	Also, enter the IP Address instead of the Host name (i.e. 155.157.123.18 for msse4hp or 155.157.123.15 for msse8sun) in the box entitled "Name or IP Address".
b. Testing:	
14. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises.** **ecs.mss.agent**	
15. From the MIB browser, query the metric and static tables.	Note: to see the metric and static tables that were registered in the configuration files.
16. Move any application and program configuration files from the configuration directory, /usr/testa/msagent2/data/TestRoot/sun5/usr/ecs/mss/MACI/cfg/OPS or ../TEST to your home directory.	To simulate missing of managed objects.
17. Wait until subagent discovers, query again the metric and static tables from the MIB browser.	(To verify ability to detect and report missing occurences of managed objects - MSS-20020, 20030)
18. Move the application and program configuration files from your home directory back to the configuration directory, /usr/testa/msagent2/data/TestRoot/sun5/usr/ecs/mss/MACI/cfg/ops or ../test	(To verify ability to discover new instances of managed objects - MSS-20010) (To verify ability to send traps/events to HPOV's trapd - MSS-36040, 36042, 36045) (To verify ability to provide an ECS management agent that is configurable to include traps to send and events to log and log file name - MSS-36060)
19. Modify the configuration files (e.g. xxxApplication.aCfg, xxxProgram.pCfg from the configuration directory, "/usr/testa/msagent2/data/TestRoot/sun5/usr/ecs/mss/MACI/cfg/ops or ../test") by changing maintenance level, version #, upgrade and downgrade version.	(To verify ability to update the object database after received a request to register/unregister a managed object - MSS-20040)
20. Wait until the subagent discovers all installed applications and programs.	(To verify ability to discover new instances of managed objects - MSS-20010)
21. Use MIB browser to see if the change takes effect.	
c. Exit:	
22. To exit this test, check the process id number of the following running jobs: **ps -ef | grep MsAg**	

23. Then, kill the master agent (MsAgAgent), the deputy (MsAgDeputy), the subagent (MsAgSubAgent) processes by executing the following command: **sudo kill -9** *<processids #>*	Note: It requires root or sudo root access to kill the master agent.
24. Check and make sure that all processes have been killed by: **ps -ef | grep MsAg**	
25. Next, make sure to clean up the CDS with this entry. You can do this by following the directions below: - Log into the HP machine where CDS resides - Log into DCE (dce_login) - Start up cdsbrowser by typing: **cdsbrowser** - Select "/.:/subsys", then "/.:subsys/HP" - Select "/.:/subsys/HP/sample-apps" - Select the object names that were registered (MsAgSubAgent and MsAgDeputy). They are: /.:/subsys/HP/sampleapps/MsAgSubAgent /.:/subsys/HP/sample-apps/MsAgDeputy - Highlight the entry and press the right button of your mouse to get a menu. Select "delete entry" to remove the entry from CDS. - Select all applications, programs, and processes under ops and test mode that were registered and delete them all: /.:/ecs/GSFC/ops/xProgram_ops /.:/ecs/GSFC/ops/xProgram_ops_group /.:/ecs/GSFC/ops/xProgram_ops_profile Remember when logging out of DCE, type kdestroy.	

4.7.2.2 Test Case 2: Polling (TC018.002)

This test case verifies the communication between the managed resources/application and the subagent to monitor the condition of its managed resources (local polling) and the management application residing on the MSS server (remote polling) and report changes.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: HpOpenView, OODCE, MACI: MsAgSubAgent, MsAgDeputy, MsAgAgent (Peer SNMP), CSS APIs (MPF)

Data: Metrics

Tools: HPOV,

Functional applications: MsAgSubAgent, MsAgDeputy, MsAgAgent, Test drivers: dgtest (deputy gate controller), mpProgram (example program that has been developed with the Managed Process Framework library. The same executable can be used to simulate multiple applications. It contains the

following capabilities: shutdown on request from the Agent, shutdown itself if start up from command line (through interactive interface), also send events if start up from command line (through interactive interface), register configuration, fault, and performance metrics. Once started, this program will read in the MSS Metric file, and connect with the subagent running on the local host (if present)).

Test Input:

none (internally)

Test Output:

Outputs include event notification when an error is detected on the managed resource or the management application and an entry into the MSS log managed by CSS EcUtLoggerRelA.

Success Criteria:

This test is successful when the polling capability detects and notifies the management application that an error condition occurred on a managed resource or a management application. Also, log entries are made on the MSS local log to reflect the error conditions detected by the subagent.

Test Procedures:

Test Case ID: TC018.002	
Test Naming: Polling	
Test Steps:	Comments:
a. Start up PEER master agent, deputy, subagent, dgtest driver, and HPOV MIB Browser:	If they are not up.
1. Enter the PEER master agent directory: **cd /usr/testa/msagent2/bin**	
2. Start the master agent in the background: **sudo MsAgAgent ../data/MsAgAgent.cfg** *<filename>* **ctrl z** **bg**	MsAgAgent - PEER master agent (COTS Application) Note: This requires root or sudo root access. Any filename is acceptable.
3. Open a window on an HP machine where MsAgDeputy resides and log into DCE by typing : **dce_login** *<username>*	MsAgDeputy - MSS Deputy (Application) This application resides on the management host, and receives events from all subagents within each DAAC. The deputy converts all events to SNMP Traps, and sends them to HPOV's trapd.
4. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the Deputy.
5. Start up the MsAgDeputy by typing: **MsAgDeputy**	In order to start the Deputy, all required environment variables must be set (as described in the installation process).
6. Create another session (on SUN or HP platform) and also log into DCE by typing: **dce_login** *<username>*	

7. Before starting up the subagent, make sure the POLL STATUS and POLL PERFORMANCE variables in /usr/testa/msagent2/data/environment.csh file is changed to a small value (e.g., 10 secs).	
8. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the SubAgent.
9. Start up the MsAgSubAgent by typing: **MsAgSubAgent ConfigFile MsAgSubAgent.aCfg ecs_mode ops**	This application has the following capabilities: - Discovery of installed/DeInstalled/Upgraded applications and programs - Management of ECS developed applications and Proxy Agents - Event notification from application to subagent - Lifecycle services (startup, shutdown) - Metric handling (performance, fault, conf) - Event Notification to MsAgDeputy (HPOV) In order to start the SubAgent, all required environment variables must be set (as described in the installation process). Once started, the SubAgent will identify all installed Applications, Programs, and previously managed Process (which are still running). An event will be generated for each identified managed object. All the entries in the Static Application and Static Program tables will be displayed prior to the subagent listening for RPC request.
10. Create a third session and also log into DCE by typing: **dce_login** *<username>*	
11. Start up dgtest test driver by typing: **dgtest** (By default, this program will connect with a subagent running on the same host the dgtest was started from) or: **dgtest** *<hostname>* (This example will connect the dgtest program to a subagent running on the host).	dgtest - deputy gate controller (test driver) Since the agent has not yet been integrated into HPOV, this test driver simulates all agent control functionality that HPOV will possess in Release A. Functionality includes: - Start & stop of Applications and Programs - Get the number of secs it will take to shutdown a Program, Process, or Application - Shut down Applications, Programs, Processes - Shut down the Subagent - Set attributes
12. Start the HpOpenView MIB browser (on the machine where MIB browser resides): **/usr/OV/bin/xnmbrowser**	
13. Enter the Host in the box entitled "Name or IP Address".	Also, enter the IP Address instead of the Host name (i.e. 155.157.123.18 for msse4hp or 155.157.123.15 for msse8sun) in the box entitled "Name or IP Address".
b. Testing:	
14. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises. ecs.mss.agent.application.appStTable.appStEntry**	
15. From the sub-branch, query appStIndex	Note: to get the index of the application to be started from the static application table.
16. From the main menu of dgtest, press 2 to startup.	

17. From the submenu of dgtest, select 1 to startup the program that has desired scenarios.	
18. Specify the index (obtained from the MIB browser) to start.	
19. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises.** **ecs.mss.agent.program.progDyTable.progDy**	Make sure the application has been started successfully.
20. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises.** **ecs.mss.agent.program.progPerfTable**	
21. From the MIB browser, query the performance metric tables.	Note: to see the performance metric tables.
22. Wait until subagent monitors.	Check events are logged to the mss.log and is notified to HPOV.
23. From the MIB browser, query again the performance metric tables.	Note: to see the performance metric tables with different scenarios (1 to 6): scen 1: min-1->min->min+1 (event) min+1->min->min-1 (no event) min-1->min->minrearm+1 (no event) scen 2: min+1->min->min-1 (event) min-1->min->minrearm+1 (no event) minrearm+1->min->min-1 (2ndevent) scen 3: max+1->max->max-1 (event) max-1->max->max+1 (no event) max+1->max->maxrearm-1(noevent) maxrearm-1->max->max+1 (event) scen4: max-1->max->max+1 (event) max+1->max->maxrearm-1 (noevent) rearmmax-1->max->max+1 (event) scen5: fault+1->fault->fault-1 (event) fault-1->fault->fault+1 (no event) fault+1->fault->faultrearm-1(noevent) faultrearm-1->fault->fault+1 (event) scen6: fault-1->fault->fault+1 (event) fault+1->fault->faultrearm-1 (noevent) rearmfault-1->fault->fault+1 (event) (To verify ability to send ECS management traps/events to the Monitor/Control service - MSS-36040) (To verify ability to provide an ECS management agent that is configurable to include traps to send and events to log and log file name - MSS-36060)
24. Compare steps 21 and 23 outputs.	See if the metric values changed in the tables. If it doesn't change, repeat steps 21 to 23. (Verify ability to detect and report change - MSS-20020, 20030)
25. Check the process corresponding to the program: **ps -ef | grep** *<programname>*	
26. Kill the process corresponding to the program: **kill -9** *<processid #>*	
27. Wait until subagent polled.	
28. From the MIB browser, query all dynamic and metric tables to verify all references to the processes and programs are gone.	(Verify ability to detect and report change - MSS-20020, 20030)
c. Exit:	

29. To exit this test, check the process id number of the following running jobs: **ps -ef | grep MsAg** **ps -ef | grep dgtest**	
30. Then, kill the master agent (MsAgAgent), the deputy (MsAgDeputy), the subagent (MsAgSubAgent), and the dgtest processes by executing the following command: **sudo kill -9** *<processids #>*	Note: It requires root or sudo root access to kill the master agent.
31. Check and make sure that all processes have been killed by: **ps -ef | grep MsAg** **ps -ef | grep dgtest**	
32. Next, make sure to clean up the CDS with this entry. You can do this by following the directions below: - Log into the HP machine where CDS resides - Log into DCE (dce_login) - Start up cdsbrowser by typing: **cdsbrowser** - Select "/.:/subsys", then "/.:subsys/HP" - Select "/.:/subsys/HP/sample-apps" - Select the object names that were registered (MsAgSubAgent and MsAgDeputy). They are: /.:/subsys/HP/sampleapps/MsAgSubAgent /.:/subsys/HP/sample-apps/MsAgDeputy - Highlight the entry and press the right button of your mouse to get a menu. Select "delete entry" to remove the entry from CDS. - Select all applications, programs, and processes under ops and test mode that were registered and delete them all: /.:/ecs/GSFC/ops/xProgram_ops /.:/ecs/GSFC/ops/xProgram_ops_group /.:/ecs/GSFC/ops/xProgram_ops_profile Remember when logging out of DCE, type kdestroy.	

4.7.3 MIB/Extensible Agent - Application 2 Thread (TC019)

This test thread verifies the full functionality of the management subagent. The subagent is primarily responsible for supporting the ECS application MIB and host MIB. Based on SNMP Get requests and MsAgDeputyGate Set requests from the master agent, they retrieve/set the appropriate MIB values.

4.7.3.1 Test Case 1: MIB Variable Validation - Get Requests (TC0019.001)

The purpose of this test case is to validate the entire set of ECS application MIB and host MIB variables through the communication between the SNMP master agent, an MSS subagent, a Host Resource agent and an ECS application to execute SNMP get requests and ensure that the MSS Management Agent Service is capable of:

- retrieving data from ECS managed objects in test or operational mode.

- communicating with the MSS Monitor/Control Service (MIB browser) to respond to requests for managed object MIB attributes.
- validating the MIB attributes of the ECS management agent for network devices.
- validating the MIB attributes of the ECS domain manager agent (master agent) which coordinates and communicates with multiple ECS management agents.
- validating the MIB attributes of the extensible ECS management agent for ECS Host systems and ECS applications.
- validating the MIB attributes of the proxy agents for ECS network devices and COTS applications that cannot be managed via SNMP.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. Sun, HP-UX), Internetworking subsystem components (i.e. LAN Network resources)

Software: HpOpenView, OODCE, MACI: MsAgSubAgent, MsAgDeputy, MsAgAgent (Peer SNMP), CSS APIs (MPF)

Data: Application MIB

Tools: HpOpenView MIB browser,

Functional applications: MsAgSubAgent, MsAgDeputy, MsAgAgent, MsAgEncps,

Test drivers: dgtest (deputy gate controller), mpProgram (example program that has been developed with the Managed Process Framework library. The same executable can be used to simulate multiple applications. It contains the following capabilities: shutdown on request from the Agent, shutdown itself if start up from command line (through interactive interface), also send events if start up from command line (through interactive interface), register configuration, fault, and performance metrics. Once started, this program will read in the MSS Metric file, and connect with the subagent running on the local host (if present)).

Test Input:

Inputs to this test case include HpOpenView commands to examine the contents of the host and application MIB.

Test Output:

Outputs include the host and application MIB variable retrieved as a result of HPOpenView get request.

Success Criteria:

This test is successful when the master agent can communicate with a host resource agent and an ECS application via SNMP Get requests to retrieve the value of a host and an application MIB variable.

Test Procedures:

Test Case ID: TC019.001	
Test Name: MIB Variable Validation-Get Request	
Test Steps:	Comments:
a. Start up PEER master agent, Encapsulator, Host Resource agent (snmpd), deputy, subagent, dgtest driver, and HPOV MIB Browser:	If they are not up.
1. Verify the snmp process is running by executing the following command on the SUN or HP-UX session: **ps -ef | grep snmpd**	Configuring and starting the Host Resource Agent on HP-UX or SUN platform.
2. If the snmpd process is listed, kill it by executing the following command: **sudo kill -9** *<processid #>*	Note: This requires root or sudo root access.
3. Enter the PEER master agent directory: **cd /usr/testa/msagent2/bin**	
4. Start the master agent in the background: **sudo MsAgAgent ../data/MsAgAgent.cfg** *<filename>* **ctrl z** **bg**	MsAgAgent - PEER master agent (COTS Application) Note: This requires root or sudo root access. Any filename is acceptable. (Verify ability to provide an extensible ECS management agent for ECS Host systems - MSS-36080)
5. Start the encapsulator in the background: **MsAgEncps -c ../data/MsAgEncps.cfg -s 1999 &**	MsAgEncps - MSS Encapsulator (COTS Application) This encapsulator enables the master agent to communicate with non-Peer SNMP agents.
6. For HP-UX host resource agent, check and make sure the snmp port number is 1161 in the /etc/services file.	
7. Start the HP-UX host resource agent: **sudo /etc/snmpd &** (This requires root or sudo root access.)	To start the Solaris host resource agent, type: sudo /opt/SUNWconn/snm/agents/snmpd -c /etc/opt/SUNWconn/snm/snmpd.conf -p 1161 &
8. Open a window on an HP machine where MsAgDeputy resides and log into DCE by typing: **dce_login** *<username>*	MsAgDeputy - MSS Deputy (Application) This application resides on the management host, and receives events from all subagents within each DAAC. The deputy converts all events to SNMP Traps, and sends them to HPOV's trapd.
9. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the Deputy.
10. Start up the MsAgDeputy by typing: **MsAgDeputy**	In order to start the Deputy, all required environment variables must be set (as described in the installation process).

11. Create another session, log into DCE by typing : **dce_login** *<username>*	
12. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the SubAgent.
13. Start up the MsAgSubAgent test driver by typing: **MsAgSubAgent ConfigFile MsAgSubAgent.aCfg ecs_mode ops**	This application has the following capabilities: - Discovery of installed/DeInstalled/Upgraded applications and programs - Management of ECS developed applications and Proxy Agents - Event notification from application to subagent - Lifecycle services (startup, shutdown) - Metric handling (performance, fault, conf) - Event Notification to MsAgDeputy (HPOV) In order to start the SubAgent, all required environment variables must be set (as described in the installation process). Once started, the SubAgent will identify all installed Applications, Programs, and previously managed Process (which are still running). An event will be generated for each identified managed object. All the entries in the Static Application and Static Program tables will be displayed prior to the subagent listening for RPC request.
14. Create another session and also log into DCE by typing: **dce_login** *<username>*	
15. Start up dgtest test driver by typing: **dgtest** (By default, this program will connect with a subagent running on the same host the dgtest was started from) or: **dgtest** *<hostname>* (This example will connect the dgtest program to a subagent running on the host).	dgtest - deputy gate controller (test driver) Since the agent has not yet been integrated into HPOV, this test driver simulates all agent control functionality that HPOV will possess in Release A. Functionality includes: - Start & stop of Applications and Programs - Get the number of secs it will take to shutdown a Program, Process, or Application - Shut down Applications, Programs, Processes - Shut down the Subagent - Set attributes
16. Start the HpOpenView MIB browser (on the machine where MIB browser resides): **/usr/OV/bin/xnmbrowser**	
17. Enter the Host name in the box entitled "Name or IP Address".	Also, enter the IP Address instead of the Host name (i.e. 155.157.123.18 for msse4hp or 155.157.123.15 for msse8sun) in the box entitled "Name or IP Address".
18. Enter "public" in the box entitled "Community Name".	
b. Testing:	

19. Go to the MIB branch entitled: **.iso.org.dod.internet.mgmt.mib-2** and query all groups in MIB II (e.g., System, Interfaces, Address Translation (AT), IP, ICMP, TCP, UDP, snmp groups)	(Verify ability to provide an ECS management agent for network devices by query AT, IP, ICMP, TCP, UDP, snmp groups…MSS-36070, 36080 (full in phase 2)).
20. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises.** **ecs.mss.agent.application.appStTable.appStEntry**	
21. From the sub-branch, query appStIndex.	Note: to get the index of the application to be started from the static application table.
22. From the main menu of dgtest, press 2 to startup.	
23. From the submenu of dgtest, select 1 to startup the application.	
24. Specify the index (obtained from the MIB browser) to start.	
25. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises.** **ecs.mss.agent** and query all tables (appStTable, appDyTable, appPerfTable, appFaultTable, appCfgTable, StprogTable, progDyTable, progFaultTable, progCfgTable, procDyTable, procPerfTable, procFaultTable).	Note: to see all values in all tables and compare these values with the values in the configuration files. Expect these values are the same. (Verify ability to retrieve data from ECS managed objects in test or operational mode - MSS-36010) (Verify ability to respond to requests for managed object MIB attributes - MSS-36020, C-HRD-32010) (Verify ability to provide an extensible ECS management for ECS applications - MSS-36090) (Verify ability to validate the MIB attributes of the proxy agents for ECS network devices and COTS applications that cannot be managed via SNMP- MSS-36100) (Verify ability to provide an ECS domain manager agent to coordinate and communicate with multiple ECS management agents - MSS-36110) (Verify ability to provide an ECS management agent that is configurable to include agent location & contact person - MSS-36060)
c. Exit:	
26. To exit this test, check the process id number of the following running jobs: **ps -ef | grep MsAg** **ps -ef | grep dgtest** **ps -ef |grep snmpd**	
27. Then, kill the master agent (MsAgAgent), the encapsulator (MsAgEncps), the deputy (MsAgDeputy), the subagent (MsAgSubAgent), the dgtest, and the snmpd processes by executing the following command: **sudo kill -9** *<processids #>*	Note: It requires root or sudo root access to kill the master agent.

28. Check and make sure that all processes have been killed by: **ps -ef | grep MsAg** **ps -ef | grep dgtest** **ps -ef | grep snmpd**	
29. Next, make sure to clean up the CDS with this entry. You can do this by following the directions below: - Log into the HP machine where CDS resides - Log into DCE (dce_login) - Start up cdsbrowser by typing: **cdsbrowser** - Select "/.:/subsys", then "/.:subsys/HP" - Select "/.:/subsys/HP/sample-apps" - Select the object names that were registered (MsAgSubAgent and MsAgDeputy). They are: /.:/subsys/HP/sampleapps/MsAgSubAgent /.:/subsys/HP/sample-apps/MsAgDeputy - Highlight the entry and press the right button of your mouse to get a menu. Select "delete entry" to remove the entry from CDS. - Select all applications, programs, and processes under ops and test mode that were registered and delete them all: /.:/ecs/GSFC/ops/xProgram_ops /.:/ecs/GSFC/ops/xProgram_ops_group /.:/ecs/GSFC/ops/xProgram_ops_profile Remember when logging out of DCE, type kdestroy.	

4.7.3.2 Test Case 2: MsAgDeputyGate Set Requests (TC019.002)

This test case verifies the communication between the MsAgDeputyGate client, an MSS subagent, an application, and the SNMP master agent to execute set requests. The MsAgDeputyGate client receives set requests from the test driver since HPOV is not available for phase 2. The MsAgDeputyGate client tells the subagent to set the value. The subagent then set the values in the applicable tables. If the attribute is a metric registered of one or more applications, all EcAgManagers for this metric will be notified to change the value. For other tables (i.e. static or application tables), EcAgManagers will not be notified of new values that are set. To conclude the test, the set value is queried with the MIB browser. The value of configuration and performance metrics is saved with the metric file.

The test also verifies that the MSS Management Agent Service is capable of:

- communicating with the MSS Monitor/Control Service to respond to requests for managed object MIB attributes.
- receiving ECS management set messages from the Monitor/Control Service.
- providing a configurable ECS management agent (set attributes).
- providing an extensible ECS management agent for ECS applications.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP), Inter-networking subsystem components (i.e. LAN Network resources)

Software: HpOpenView, OODCE, MACI: MsAgSubAgent, MsAgDeputy, MsAgAgent (Peer SNMP), CSS APIs (MPF)

Data: N/A

Tools: HpOpenView MIB browser,

Functional applications: MsAgSubAgent, MsAgDeputy, MsAgAgent, Test drivers: dgtest (deputy gate controller), mpProgram (example program that has been developed with the Managed Process Framework library. The same executable can be used to simulate multiple applications. It contains the following capabilities: shutdown on request from the Agent, shutdown itself if start up from command line (through interactive interface), also send events if start up from command line (through interactive interface), register configuration, fault, and performance metrics. Once started, this program will read in the MSS Metric file, and connect with the subagent running on the local host (if present)).

Test Input:

Inputs to this test case include all values which can be set.

Test Output:

MIB variable retrieved as a result of HpOpenView set request.

Success Criteria:

This test is successful when the master agent can communicate with an application via MsAgDeputyGate set and SNMP Get requests to set/retrieve the value of a MIB variable.

Test Procedures:

Test Case ID: TC019.002	
Test Name: MsAgDeputyGate Set Request	
Test Steps:	Comments:
a. Start up PEER master agent, deputy, subagent, dgtest driver, and HPOV MIB Browser:	If they are not up.
1. Enter the PEER master agent directory: ***cd /usr/testa/msagent2/bin***	
2. Start the master agent in the background: **sudo MsAgAgent ../data/MsAgAgent.cfg** *<filename>* **ctrl z** **bg**	MsAgAgent - PEER master agent (COTS Application) Note: This requires root or sudo root access. Any filename is acceptable.

3. Open a window on an HP machine where MsAgDeputy resides and log into DCE by typing : **dce_login** *<username>*	MsAgDeputy - MSS Deputy (Application) This application resides on the management host, and receives events from all subagents within each DAAC. The deputy converts all events to SNMP Traps, and sends them to HPOV's trapd.
4. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the Deputy.
5. Start up the MsAgDeputy by typing: **MsAgDeputy**	In order to start the Deputy, all required environment variables must be set (as described in the installation process).
6. Create another session (on SUN or HP platform) and also log into DCE by typing: **dce_login** *<username>*	
7. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the SubAgent.
8. Start up the MsAgSubAgent by typing: **MsAgSubAgent ConfigFile MsAgSubAgent.aCfg ecs_mode ops**	This application has the following capabilities: - Discovery of installed/DeInstalled/Upgraded applications and programs - Management of ECS developed applications and Proxy Agents - Event notification from application to subagent - Lifecycle services (startup, shutdown) - Metric handling (performance, fault, conf) - Event Notification to MsAgDeputy (HPOV) In order to start the SubAgent, all required environment variables must be set (as described in the installation process). Once started, the SubAgent will identify all installed Applications, Programs, and previously managed Process (which are still running). An event will be generated for each identified managed object. All the entries in the Static Application and Static Program tables will be displayed prior to the subagent listening for RPC request.
9. Create a third session and also log into DCE by typing: **dce_login** *<username>*	
10. Start up dgtest test driver by typing: **dgtest** (By default, this program will connect with a subagent running on the same host the dgtest was started from) or: **dgtest** *<hostname>* (This example will connect the dgtest program to a subagent running on the host).	dgtest - deputy gate controller (test driver) Since the agent has not yet been integrated into HPOV, this test driver simulates all agent control functionality that HPOV will possess in Release A. Functionality includes: - Start & stop of Applications and Programs - Get the number of secs it will take to shutdown a Program, Process, or Application - Shut down Applications, Programs, Processes - Shut down the Subagent - Set attributes
11. Start the HpOpenView MIB browser (on the machine where MIB browser resides): **/usr/OV/bin/xnmbrowser**	

12. Enter the Host in the box entitled "Name or IP Address".	Also, enter the IP Address instead of the Host name (i.e. 155.157.123.18 for msse4hp or 155.157.123.15 for msse8sun) in the box entitled "Name or IP Address".
b. Testing:	
13. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises.ecs.mss.agent**	Note: To get the Index of the table entry of all tables.
14. Create another session (open another window in an x-term) and also log into DCE by typing: **dce_login** *<username>*	
15. Start up dgtest test driver by typing: **dgtest**	
16. From the main menu of dgtest, enter 6 for set attribute	
17. From the submenu of set attribute, enter table ID # (0 for static appl, 2 for appl perf, 4 for appl conf, 5 for static prog, 8 for prog conf, 9 for dyn process, 10 for process perf)	
18. From the submenu of set attribute, enter Table Index value (e.g., indexes 0, 1, 2, 3, 4, 5, 6…).	
19. From the submenu of set attribute (which Index to change), enter Attribute Index value.	Following are Attribute Index of tables that can be set: StAppTable: 0 for StAppEventLevel 1 for StAppLogLevel 2 for StAppLogMax StProgramTable (5): 0 for StProgTimeOut DynProcTable (9): 0 for TeDyProcTimeToStop MsEAgAppPerfTbl (2) & MsEAgProcPerfTbl (10): 0 for MsEAgTePerfFaultThreshold 1 for MsEAgTePerfRearmFaultThreshold 2 for MsEAgTePerfMaxThreshold 3 for MsEAgTePerfRearmMaxThreshold 4 for MsEAgTePerfMinThreshold 5 for MsEAgTePerfRearmMinThreshold MsEAgAppConfigTbl (4) & MsEAgProgConfigTbl (8): 0 for MsEAgTeConfigValue
20. From the MIB browser, query the change value table.	To see values changed. (Verify ability to respond to requests for managed object MIB attributes - MSS-36020) (Verify ability to receive ECS management set message using a reliable and secure mechanism - MSS-36050, 36052, 36055) (Verify ability to provide an ECS management agent that is configurable to include community to respond to and set attributes - MSS-36060) (Verify ability to provide an extensible ECS management agent for ECS applications - MSS36090)
c. Exit:	

21. To exit this test, check the process id number of the following running jobs: **ps -ef | grep MsAg** **ps -ef | grep dgtest**	
22. Then, kill the master agent (MsAgAgent), the deputy (MsAgDeputy), the subagent (MsAgSubAgent), and the dgtest processes by executing the following command: **sudo kill -9** *<processids #>*	Note: It requires root or sudo root access to kill the master agent.
23. Check and make sure that all processes have been killed by: **ps -ef | grep MsAg** **ps -ef | grep dgtest**	
24. Next, make sure to clean up the CDS with this entry. You can do this by following the directions below: - Log into the HP machine where CDS resides - Log into DCE (dce_login) - Start up cdsbrowser by typing: **cdsbrowser** - Select "/.:/subsys", then "/.:subsys/HP" - Select "/.:/subsys/HP/sample-apps" - Select the object names that were registered (MsAgSubAgent and MsAgDeputy). They are: /.:/subsys/HP/sampleapps/MsAgSubAgent /.:/subsys/HP/sample-apps/MsAgDeputy - Highlight the entry and press the right button of your mouse to get a menu. Select "delete entry" to remove the entry from CDS. - Select all applications, programs, and processes under ops and test mode that were registered and delete them all: /.:/ecs/GSFC/ops/xProgram_ops /.:/ecs/GSFC/ops/xProgram_ops_group /.:/ecs/GSFC/ops/xProgram_ops_profile Remember when logging out of DCE, type kdestroy.	

4.7.4 Process Framework Thread (TC040)

The purpose of this thread is to verify the functionality of the Process Framework thread. Process framework is an extensible mechanism for ECS client and server applications to transparently utilize ECS infrastructure features including OODCE security management, OODCE naming/directory service, Message Passing, FTP, Server Request Framework, and Lifecycle Management. The purpose of Process Framework is to encapsulate implementation details of ECS infrastructure services and removes the need for programmers to rewrite common initialization code. It will also ensure design and implementation consistency for all ECS Client and Server Applications and provide a basis for future extensions to infrastructure mechanisms without adversely affecting the ECS developers.

4.7.4.1 Test Case 1: Basic Process Functionality - Configuration File (TC040.001)

The purpose of this test is to verify that the Process Framework will maintain basic process information needed for most ECS applications. Specifically, this test demonstrates that Process Framework maintains the following information supplied by a configuration file:

> Mode, DeltaTime, Site Name, Executable Name, Process ID, Application ID, Program ID, Major Version, Minor Version.

Test Configuration:

Hardware: HP and Sun workstations/servers HCI, Internetworking subsystem CI (LAN Network resources)-INCI

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions

Data: N/A

Tools: bp_sun, bp_hp, ConfigFile

Test Input:

The configuration file containing the desired variable values must be passed as a command line argument to the test driver along with the desired mode of operation.

Test Output:

The client test driver should issue statements indicating the values stored in the Process Framework for the variables given above.

Success Criteria:

The values placed in the configuration file match those returned in the test driver.

Test Case ID: TC040.001	
Test Name: Basic Process Functionality-Configuration File	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.
2. Start the test application by typing the following in the test machine window: **bp_hp ConfigFile** *<configuration file filename>* **ecs_mode** ***<mode>***	The values in this configuration file are the desired values for this testcase. Process Framework should store these values and associate them with the test driver process. **ecs_mode** is the desired mode to run the application in. C-CSS-27010

3. Verify a message indicating the mode of operation is displayed in the client window.	C-CSS-27050
4. Verify a message indicating the delta time is displayed in the client window.	C-CSS-27050
5. Verify a message indicating the executable name is displayed in the client window.	C-CSS-27050
6. Verify a message indicating the process ID is displayed in the client window.	C-CSS-27050
7. Verify a message indicating the Application ID is displayed in the client window.	C-CSS-27050
8. Verify a message indicating the Program ID is displayed in the client window.	C-CSS-27050
9. Verify a message indicating the Major Version is displayed in the client window.	C-CSS-27050
10. Verify a message indicating the Minor Version is displayed in the client window.	C-CSS-27050

4.7.4.2 Test Case 2: Basic Process Functionality - Command Line (TC040.002)

The purpose of this test is to verify that the Process Framework will maintain basic process information needed for most ECS applications. Specifically, this test demonstrates that Process Framework will override information contained in a configuration file with values placed on the executable's command line.

Test Configuration:

Hardware: HP and Sun workstations/servers HCI, Internetworking subsystem CI (LAN Network resources)-INCI

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions

Data: N/A

Tools: bp_sun, bp_hp, ConfigFile

Test Input:

The configuration file containing the desired variable values must be passed as a command line argument to the test driver along with the desired mode of operation.

Test Output:

The client test driver should issue statements indicating the values stored in the Process Framework for the variables given above except for the process Id variable which is overridden on the command line.

Success Criteria:

The program exits when the proper command line arguments are not supplied. The process id value provided on the command line is used by Process Framework instead of the value in the configuration file. All other values placed in the configuration file match those returned in the test driver.

Test Case ID: TC040.002	
Test Name: Basic Process Functionality-Configuration File	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.
2. Start the test application by typing the following in the test machine window: **bp_hp ConfigFile** *<configuration file filename>*	The values in this configuration file are the desired values for this testcase. Process Framework should store these values and associate them with the test driver process. C-CSS-27010
3. Verify a message indicating the appropriate arguments were not supplied is displayed and the application exits.	The name of the configuration file and the mode must be specified on the command line. C-CSS-27040
4. Start the test application by typing the following in the test machine window: **bp_hp ecs_mode *<mode>***	**ecs_mode** is the desired mode to run the application in.
5. Verify a message indicating the appropriate arguments were not supplied is displayed and the application exits.	The name of the configuration file and the mode must be specified on the command line. C-CSS-27040
6. Start the test application by typing the following in the test machine window: **bp_hp ConfigFile** *<configuration file filename>* **ecs_mode *<mode>*** **program id** *<value other than that in the configuration file>*	The values in this configuration file are the desired values for this testcase. Process Framework should store these values and associate them with the test driver process. **ecs_mode** is the desired mode to run the application in. The **program id** variable placed in the command line should override that placed in the configuration file. C-CSS-27020
7. Verify a message indicating the mode of operation is displayed in the client window.	C-CSS-27050
8. Verify a message indicating the delta time is displayed in the client window.	C-CSS-27050
9. Verify a message indicating the executable name is displayed in the client window.	C-CSS-27050
10. Verify a message indicating the process ID is displayed in the client window.	C-CSS-27050
11. Verify a message indicating the Application ID is displayed in the client window.	C-CSS-27050
12. Verify a message indicating the Program ID is displayed in the client window.	This value should correspond to the value placed on the command line, not the value specified in the configuration file. C-CSS-27030, C-CSS-27050
13. Verify a message indicating the Major Version is displayed in the client window.	C-CSS-27050
14. Verify a message indicating the Minor Version is displayed in the client window.	C-CSS-27050

4.7.4.3 Test Case 3: Server Functionality (TC040.003)

The purpose of this test is to verify that Process Framework provides interfaces to COTS and custom developed APIs implementing general server functionality. Specifically, this test will demonstrate that Process Framework will allow the following attributes to be set for a server application:

- Server short name
- Profile Name
- Group Name
- Server principal name
- Keytab file name
- ACL database filename (or sybase key)
- Protocol policy
- Host Policy

Test Configuration:

Hardware: HP and Sun workstations/server HCI, Inter networking subsystem CI (LAN Network resources), INCI

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions

Data: N/A

Tools: pf_cli_sun, pf_srv_sun, pf_cli_hp, pf_srv_hp, ConfigFile

Test Input:

The configuration file containing the desired variable values must be passed as a command line argument to the server test driver as well as the desired mode. The client needs the CDS entry name for the server.

Test Output:

The server test driver should issue statements indicating the values stored in the Process Framework for the variables given above.

Success Criteria:

The values placed in the configuration file match those returned in the test driver. The client is able to connect to the server.

Test Case ID: TC040.003	
Test Name: Server Functionality	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.
2. Create two sessions (i.e. open two xterm windows) and log into DCE in both by typing : **dce_login** *username password*	
3. In one window, start the cdsbrowser application in the background by typing: **cdsbrowser &**	Note the cdsbrowser application is available only on HP platforms. When executing this testcase on other platforms create a third session in step 2. In this window log into a HP machine in the same cell as the test machine, set your DISPLAY variable to point back to the test machine, and then execute step 3.
4. Start the test application by typing the following in the test machine window: **pf_srv_hp ConfigFile** *<configuration file filename>* **ecs_mode** ***<mode>***	The values in this configuration file are the desired values for this testcase. Process Framework should store these values and associate them with the test driver process. **ecs_mode** is the desired mode to run the application in. C-CSS-27010
5. Using the cdsbrowser, verify that an entry has been added to CDS for the server short name specified in the configuration file.	C-CSS-27060
6. Using the cdsbrowser, verify that an entry has been added to CDS for the server group name specified in the configuration file.	C-CSS-27060
7. Using the cdsbrowser, verify that an entry has been added to CDS for the server profile name specified in the configuration file.	C-CSS-27060
8. Verify a message indicating the protocol policy is displayed in the server window.	C-CSS-27090
9. Verify a message indicating the host policy is displayed in the server window.	C-CSS-27100
10. Verify a message indicating the server principal name is displayed in the server window.	The server principal name must be a name located in the DCE registry. C-CSS-27080
11. Verify a message indicating the keytab filename is displayed in the server window.	An entry for the principal name displayed in step 10 must be included in this file. C-CSS-27080
12. Verify a message indicating the ACL database filename or sybase key is displayed in the server window.	The tester executing this test must be granted access to this server via inclusion in this ACL database. C-CSS-27080
13. In the second window start the client application by typing the following: **pf_cli_hp** *<cds entry name>*	The cds entry name should correspond to the entry the server was registered in the CDS in step 5.
14. Verify a message indicating the client successfully contacted the server is displayed in the client window.	Since the client was able to contact the server then the server was able to establish its identity in the DCE Registry. C-CSS-27070
15. Kill the server process by issuing a kill signal in the server window. (e.g. **Ctrl-C)**	

16. Remove the server object from the CDS namespace by highlighting the server object in the cdsbrowser and clicking on the **options->delete** entry button in the toolbar.	
17. In each window logout of dce by issuing the following commands: **kdestroy** **exit**	

4.7.4.4 Test Case 4: Services Interface Functionality (TC040.004)

The purpose of this test is to verify that Process Framework provides interfaces to the FTP batch processing facility, Management Agent Framework, the Message Passing Service and the Server Request Framework.

Test Configuration:

Hardware: HP and Sun workstations/server HCI, Inter networking subsystem CI (LAN Network resources)-INCI

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions, ECS Common library functions.

Data: N/A

Tools: serv_cli_sun, susp_cli_sun, pf2_srv_sun, serv_cli_hp, susp_cli_hp pf2_srv_hp, ConfigFile

Test Input:

The configuration file containing the desired variable values must be passed as a command line argument to the server test driver as well as the desired mode. The client applications need the CDS entry name for the server.

Test Output:

The server test driver should issue statements indicating the values stored in the Process Framework. The client test drivers will issue statements indicating messages have been sent via the Asynchronous Message Passing Service and the Server Request Framework, a FTP batch process has been scheduled and completed, and the server has been shutdown.

Success Criteria:

The values placed in the configuration file match those returned in the test driver. Messages are sent, a FTP batch process is completed, and the server is shutdown.

Test Procedures:

Test Case ID: TC040.004	
Test Name: Services Interfaces Functionality	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.
2. Create three sessions (i.e. open three xterm windows) and log into DCE in each by typing : **dce_login** *username password*	
3. In one window, start the cdsbrowser application in the background by typing: **cdsbrowser &**	Note the cdsbrowser application is available only on HP platforms. When executing this testcase on other platforms create a third session in step 2. In this window log into a HP machine in the same cell as the test machine, set your DISPLAY variable to point back to the test machine, and then execute step 3.
4. Start the test server application by typing the following in a machine window: **pf2_srv_hp ConfigFile** *<configuration file filename>* **ecs_mode *<mode>***	The values in this configuration file are the desired values for this testcase. Process Framework should store these values and associate them with the test driver process. **Ecs_mode** is the desired mode to run the application in. C-CSS-27010
5. Using the cdsbrowser, verify that an entry has been added to CDS for the server short name specified in the configuration file.	C-CSS-27060
6. Using the cdsbrowser, verify that an entry has been added to CDS for the server group name specified in the configuration file.	C-CSS-27060
7. Using the cdsbrowser, verify that an entry has been added to CDS for the server profile name specified in the configuration file.	C-CSS-27060
8. Verify a message indicating the protocol policy is displayed in the server window.	C-CSS-27090
9. Verify a message indicating the host policy is displayed in the server window.	C-CSS-27100
10. Verify a message indicating the server principal name is displayed in the server window.	The server principal name must be a name located in the DCE registry. C-CSS-27080
11. Verify a message indicating the keytab filename is displayed in the server window.	An entry for the principal name displayed in step 10 must be included in this file. C-CSS-27080
12. Verify a message indicating the ACL database filename or sybase key is displayed in the server window.	The tester executing this test must be granted access to this server via inclusion in this ACL database. C-CSS-27080
13. In the second window start the services client application by typing the following: **serv_cli_hp** *<cds entry name>*	The cds entry name should correspond to the entry the server was registered in the CDS in step 5.
14. Verify a message indicating the client connected to the server appears in the services client window.	Since the client was able to contact the server then the server was able to establish its identity in the DCE Registry. C-CSS-27070

15. Verify a message indicating the client has successfully sent a message to an arbitrary receiver appears in the services client window.	This is done via the Asynchronous Message Passing interface. C-CSS-27140
16. Verify a message indicating the server has sent a message to itself appears in the server window	This done via the Server Request Framework interface. C-CSS-27130
17. Verify a message indicating the server has received the message it sent to itself appears in the server window.	
18. Verify a message indicating the client has successfully completed an FTP batch process appears in the services client window.	C-CSS-27120
19. In the third window start the shutdown client application. **susp_cli_hp** *<cds entry name>*	
20. Select the option to shutdown the application.	
21. In the second window restart the services client application by typing the following: **serv_cli_hp** *<cds entry name>*	The cds entry name should correspond to the entry the server was registered in the CDS in step 5.
22. Verify a message indicating the client could not find the server and it is now exiting appears in the services client window.	This verifies that the server has been shutdown. C-CSS-27150
23. Kill the server process by issuing a kill signal in the server window. (e.g. **Ctrl-C)**	
24. Remove the server object from the CDS namespace by highlighting the server object in the cdsbrowser and clicking on the **options->delete** entry button in the toolbar.	
25. In each window logout of dce by issuing the following commands: **kdestroy** **exit**	

4.7.4.5 Test Case 5: Error Recovery (TC040.005)

The purpose of this test is to verify that if an error condition arises, Process Framework properly logs the error and continues in its previous thread of execution.

Test Configuration:

Hardware: HP and Sun workstations/server HCI, Inter networking subsystem CI (LAN Network resources)-INCI

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions, ECS Common library functions.

Data: N/A

Tools: pfrc_sun, pfrc_hp, pfrs_sun, pfrs_hp, config_file

Test Input:

The configuration file containing the desired variable values must be passed as a command line argument to the server test driver as well as the desired mode. The client applications need the CDS entry name for the server.

Test Output:

The server test driver should issue statements indicating the values stored in the Process Framework. The server test drivers will issue statements indicating errors occurred while using one of the provided interfaces. Messages will then appear indicating that the interrupted transaction was completed.

Success Criteria:

The error messages are reported and the server is able to recover and complete transaction.

Test Procedures:

Test Case ID: TC040.004	
Test Name: Services Interfaces Functionality	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.
2. Create two sessions (i.e. open two xterm windows) and log into DCE in each by typing : **dce_login** *username password*	
3. In one window, start the cdsbrowser application in the background by typing: **cdsbrowser &**	Note the cdsbrowser application is available only on HP platforms. When executing this testcase on other platforms create a third session in step 2. In this window log into a HP machine in the same cell as the test machine, set your DISPLAY variable to point back to the test machine, and then execute step 3.
4. Start the test server application by typing the following in a machine window: **pfrs_hp ConfigFile** *<configuration file filename>* **ecs_mode** ***<mode>***	The values in this configuration file are the desired values for this testcase. Process Framework should store these values and associate them with the test driver process. **Ecs_mode** is the desired mode to run the application in. C-CSS-27010
5. In the second window start the services client application by typing the following: **pfrc_cli_hp** *<cds entry name>*	The cds entry name should correspond to the entry the server was registered in the CDS in step 4.
6. In the client window select which service to use: • Message Passsing • FTP Batch Processing • Server Request Framework	
7. Kill the server process by issuing a kill signal in the server window. (e.g. **Ctrl-C)**	

8. Verify a message appears in client window indicating the logging of the error that occurred.	C-CSS-27110
9. In the server window restart the server application.	
10. Verify a message indicating the server has logged the earlier failure and is proceeding with the previous transaction.	C-CSS-27110
11. Verify a message indicating the previous transaction has completed successfully appears in the server window.	The transaction will only be completed if the chosen interface supports persistance and recovery (e.g. Message Passing).
12. Repeat steps 6-11 for each service.	

4.7.5 Application Management Services Build (BC006)

The Application Management Services build is a collection of the Management Process Framework thread, the Lifecycle Services thread, the Startup/Shutdown thread, the MIB/Extensible Agent - Application 2 thread and the External Interface Communication Build. The External Interface Communication build is an aggregation of the User Account Management, FTP/kFTP, Gateway Client/Server, and Security Authorization threads. The functions to be tested consist of some of the key functionality's of these threads and builds.

The objective of this test is to verify that the previously tested functionality of the threads and builds remains intact following integration.

4.7.5.1 Test Case 1: Application Management Services Test (BC006.001)

The Application Management Services test verifies the overall functionality of the MSS Lifecycle Services Thread, Discover/Monitor Thread, MIB/Extensible Agent - Application 2 Thread and Managed Process Framework Thread. The purpose of this test is to verify that MSS agent using the Process Framework is able to support ECS client and server applications by transparently including ECS infrastructure features such as OODCE security management, OODCE naming/directory service, Message Passing, FTP and Lifecycle Management (i. e. initialize the process application and infrastructure in a consistent way and provide some basic process information, interface to Mode Management, interface to Error-Event Logging, support to Life Cycle management…). This test also verify the startup/shutdown of an application/program/process, the discovery and report of new instances of managed objects, the monitoring of the condition of its managed resources and managed application. To ensure that the application MIB and host MIB can be retrieved and certain values of application MIB can also be set.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP), Inter-networking subsystem components (i.e. LAN Network resources)

Software: HpOpenView, DCE/OODCE, MsAgSubAgent, MsAgDeputy, MsAgAgent (Peer SNMP), EcAgManager, DNS, CSS APIs (MPF), library functions, NCSA HTTPD server

Data: HTTPDProxy.AppCfg, HTTPDProxy.EcAgCfg, metrics, configuration files created for applications and programs

Tools: HpOpenView, WWW browser

Test Input:

Input to this test case include configuration files, HpOpenView commands to examine the contents of the host MIB, Peer Network configured as the SNMP master agent, an ECS subagent, an ECS application, program, process, a management agent, a management application, and a managed resource.

Test Output:

Output to this test case include the following:

- MIB contents displayed on HpOpenView screen
- MIB variable retrieved as a result of HpOpenView get request
- MIB variable set as a result of HpOpenView set request
- Event notifications to the management agent and log entries to the MSS log are reported
- Notification that an application, a program, and a COTS product are started up and shutdown, and that a process is shutdown

Success Criteria:

This test is successful when MIB values displayed via HpOpenView match those specified in the MIB definition documentation and the master agent can communicate with an ECS application via SNMP Get/Set requests to retrieve and set the value of a MIB variable. Discovery can discover and report new instances, missing occurrences of managed objects. Polling can detect and notify the management application when an error occurred on a managed resource or management application. Also, log entries are made on the MSS local log to reflect the error conditions detected by the subagent. An application, a program, a COTS product and a subagent can be started and shutdown, a process can be shutdown.

Test Procedures:

Test Case ID: BC006.001	
Test Name: Application Management Services Test	
Test Steps:	Comments:
a. Start up PEER master agent, Encapsulator, Host Resource agent (snmpd), deputy, subagent, dgtest driver, and HPOV MIB Browser:	If they are not up.
1. Verify the snmp process is running by executing the following command on the SUN or HP-UX session: **ps -ef | grep** *<snmpd>*	Configuring and starting the Host Resource Agent on HP-UX or SUN platform.
2. If the snmpd process is listed, kill it by executing the following command: **sudo kill -9 <***processid #>*	Note: This requires root or sudo root access.
3. Enter the PEER master agent directory: **cd /usr/testa/msagent2/bin**	
4. Start the master agent in the background: **sudo ./MsAgAgent ../data/MsAgAgent.cfg filename** **ctrl z** **bg**	MsAgAgent - PEER master agent (COTS Application) Note: This requires root or sudo root access. Any filename is acceptable.
5. Start the encapsulator in the background: **MsAgEncps -c ../data/MsAgEncps.cfg -s 1999 &**	MsAgEncps - MSS Encapsulator (COTS Application) This encapsulator enables the master agent to communicate with non-Peer SNMP agents.
6. For HP-UX host resource agent, check and make sure the snmp port number is 1161 in the /etc/services file.	
7. Start the HP-UX host resource agent: **sudo /etc/snmpd &** (This requires root or sudo root access.)	To start the **Solaris** host resource agent, type: sudo /opt/SUNWconn/snm/agents/snmpd -c /etc/opt/SUNWconn/snm/snmpd.conf -p 1161 &
8. Open a window on an HP machine where MsAgDeputy resided and log into DCE by typing: **dce_login** *<username>*	MsAgDeputy - MSS Deputy (Application) This application resides on the management host, and receives events from all subagents within each DAAC. The deputy converts all events to SNMP Traps, and sends them to HPOV's trapd.
9. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the Deputy.
10. Start up the MsAgDeputy by typing: **MsAgDeputy**	In order to start the Deputy, all required environment variables must be set (as described in the installation process).
11. Create another session, log into DCE by typing : **dce_login** *<username>*	
12. Source the environment.csh file: **source ../data/environment.csh**	This will set up all the appropriate environment variables needed by the SubAgent.

13. Start up the MsAgSubAgent test driver by typing: **MsAgSubAgent ConfigFile MsAgSubAgent.aCfg ecs_mode ops**	This application has the following capabilities: - Discovery of installed/DeInstalled/Upgraded applications and programs - Management of ECS developed applications and Proxy Agents - Event notification from application to subagent - Lifecycle services (startup, shutdown) - Metric handling (performance, fault, conf) - Event Notification to MsAgDeputy (HPOV) In order to start the SubAgent, all required environment variables must be set (as described in the installation process). Once started, the SubAgent will identify all installed Applications, Programs, and previously managed Process (which are still running). An event will be generated for each identified managed object. All the entries in the Static Application and Static Program tables will be displayed prior to the subagent listening for RPC request.
14. Create another session and also log into DCE by typing: **dce_login** *<username>*	
15. Start up dgtest test driver by typing: **dgtest** (By default, this program will connect with a subagent running on the same host the dgtest was started from) or: **dgtest** *<hostname>* (This example will connect the dgtest program to a subagent running on the host).	dgtest - deputy gate controller (test driver) Since the agent has not yet been integrated into HPOV, this test driver simulates all agent control functionality that HPOV will possess in Release A. Functionality includes: - Start & stop of Applications and Programs - Get the number of secs it will take to shutdown a Program, Process, or Application - Shut down Applications, Programs, Processes - Shut down the Subagent - Set attributes
16. Start the HpOpenView MIB browser (on msse4hp machine): **/usr/OV/bin/xnmbrowser**	
17. Enter the Host name or the IP Address in the box entitled "Name or IP Address".	
18. Enter "public" in the box entitled "Community Name".	
b. Testing:	
19. Go to the MIB branch entitled: **.iso.org.dod.internet.mgmt.mib-2** and query all groups in MIB II (e.g., System, Interfaces, Address Translation (AT), IP, ICMP, TCP, UDP, snmp groups)	Verify ability to provide an ECS management agent for network devices by query AT, IP, ICMP, TCP, UDP, snmp groups.
20. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises. ecs.mss.agent.application.appStTable.appStEntry**	
21. From the sub-branch, query appStIndex.	Note: to get the index of the application to be startup from the static application table.
22. From the main menu of dgtest, press 2 to startup.	

23. From the submenu of dgtest, select 1 to startup the application.	
24. Specify the index (obtained from the MIB browser) to start.	
25. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises.ecs.mss.agent** and query all tables (appStTable, appDyTable, appPerfTable, appFaultTable, appCfgTable, progTable, progDyTable, progFaultTable, progCfgTable, procDyTable, procPerfTable, procFaultTable).	Note: to see all values in all tables and compare these values with the values in the configuration files. Expect these values are the same. Also, to get the index of the table entry of all tables in order to set values.
26. Repeat steps 20 to 24 for the following scenario: - startup application 1 - startup application 2 - startup application 3 - startup application 4 - shutdown application 2 - shutdown application 3 - shutdown program 1 of application 1 - shutdown a process of application 1 - kill a process of application 4 - shutdown application 4	all 4 applications are up. All references to application 2 and 3, their programs and processes are gone. Program 1 and its process are gone. The process and program of application 1 is gone. The process and program of application 4 is gone. The remaining 2 programs and processes of application 4 are gone.
27. **Proxy Agent……**	
28. From the main menu of dgtest, enter 6 for set attribute	
27. From the submenu of set attribute, enter table ID # (0 for static appl, 2 for appl perf, 4 for appl conf, 5 for static prog, 8 for prog conf, 9 for dyn process, 10 for process perf)	
28. From the submenu of set attribute, enter Table Index value obtained from step 25 (e.g., indexes 0, 1, 2, 3, 4, 5, 6…).	
29. From the submenu of set attribute (which Index to change), enter Attribute Index value.	Following are Attribute Index of tables that can be set: • StAppTable (0): 0 for StAppEventLevel 1 for StAppLogLevel 2 for StAppLogMax • StProgramTable (5): 0 for StProgTimeOut • DynProcTable (9): 0 for TeDyProcTimeToStop • MsEAgAppPerfTbl (2) & MsEAgProcPerfTbl (10): 0 for MsEAgTePerfFaultThreshold 1 for MsEAgTePerfRearmFaultThreshold 2 for MsEAgTePerfMaxThreshold 3 for MsEAgTePerfRearmMaxThreshold 4 for MsEAgTePerfMinThreshold 5 for MsEAgTePerfRearmMinThreshold • MsEAgAppConfigTbl (4) & MsEAgProgConfigTbl (8): 0 for MsEAgTeConfigValue

30. From the MIB browser, query the change value table.	To see values changed.
31. From the MIB browser, query the metric and static tables.	Note: to see the metric and static tables that were registered in the configuration files.
32. Move any application and program configuration files from the configuration directory, /usr/testa/msagent2/data/TestRoot/sun5/usr/ecs/mss/MACI/cfg/ops or ../test to your home directory.	To simulate missing of managed objects.
33. Wait until subagent discovers, query again the metric and static tables from the MIB browser.	
34. Move the application and program configuration files from your home directory back to the configuration directory, /usr/testa/msagent2/data/TestRoot/sun5/usr/ecs/mss/MACI/cfg/ops or ../test	
35. Modify the configuration files (e.g. xxxApplication.aCfg, xxxProgram.pCfg from the configuration directory, "/usr/testa/msagent2/data/TestRoot/sun5/usr/ecs/mss/MACI/cfg/ops or ../test") by changing maintenance level, version #, upgrade and downgrade version.	
36. Wait until the subagent discovers all installed applications and programs.	
37. Use MIB browser to see if the change takes effect.	
38. From the main menu of dgtest, press 2 to startup.	
39. From the submenu of dgtest, select 1 to startup the program that has desired scenarios.	
40. Specify the index (obtained from the MIB browser) to start.	
41. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises.ecs.mss.agent.program.progDyTable.progDy**	Make sure the application has been started successfully.
42. Go to the MIB branch entitled: **.iso.org.dod.internet.private.enterprises.ecs.mss.agent.program.progPerfTable**	
43. From the MIB browser, query the performance metric tables.	Note: to see the performance metric tables.
44. Wait until subagent monitors.	
45. From the MIB browser, query again the performance metric tables.	Note: to see the performance metric tables with different scenarios (1 to 6): scen 1: min-1->min->min+1 (event) min+1->min->min-1 (no event) min-1->min->minrearm+1 (no event) scen 2: min+1->min->min-1 (event) min-1->min->minrearm+1 (no event) minrearm+1->min->min-1 (2ndevent) scen 3: max+1->max->max-1 (event) max-1->max->max+1 (no event) max+1->max->maxrearm-1(noevent) maxrearm-1->max->max+1 (event) scen4: max-1->max->max+1 (event)

	max+1->max->maxrearm-1 (noevent) rearmmax-1->max->max+1 (event) scen5: fault+1->fault->fault-1 (event) fault-1->fault->fault+1 (no event) fault+1->fault->faultrearm-1(noevent) faultrearm-1->fault->fault+1 (event) scen6: fault-1->fault->fault+1 (event) fault+1->fault->faultrearm-1 (noevent) rearmfault-1->fault->fault+1 (event)
46. Compare steps 43 and 45 outputs.	See if the metric values changed in the tables. If it doesn't change, repeat steps 43 to 45.
47. Check the process corresponding to the program: **ps -ef | grep** *<programname>*	
48. Kill the process corresponding to the program: **kill -9** *<processid #>*	
49. Wait until subagent polled.	
50. From the MIB browser, query all dynamic and metric tables to verify all references to the processes and programs are gone.	
51. From the main menu of dgtest, press 7 to shutdown the subagent.	
52. Verify the subagent is gone by executing the following command: **ps -ef | grep** *<SubAgentprocessid #>*	
c. Exit:	
. To exit this test, check the process id number of the following running jobs: **ps -ef | grep MsAg** **ps -ef | grep dgtest** **ps -ef | grep snmpd**	
27. Then, kill the master agent (MsAgAgent), the encapsulator (MsAgEncps), the deputy (MsAgDeputy), the subagent (MsAgSubAgent), the dgtest, and the snmpd processes by executing the following command: **sudo kill -9** *<processids #>*	Note: It requires root or sudo root access to kill the master agent.
28. Check and make sure that all processes have been killed by: **ps -ef | grep MsAg** **ps -ef | grep dgtest** **ps -ef | grep snmpd**	
29. Next, make sure to clean up the CDS with this entry. You can do this by following the directions below: - Log into the HP machine where CDS resides - Log into DCE (dce_login) - setenv DISPLAY machineaddress:0.0 - Start up cdsbrowser by typing: **cdsbrowser** - Select "/.:/subsys", then "/.:subsys/HP" - Select "/.:/subsys/HP/sample-apps" - Select the object names that were registered (MsAgSubAgent and MsAgDeputy). They are: /.:/subsys/HP/sampleapps/MsAgSubAgent /.:/subsys/HP/sample-apps/MsAgDeputy	EOSD0510#A, EOSD0740#A, EOSD0750#A, EOSD0500#A, ESN-0790#A, ESN-0005#A, ESN-0840#A, ESN-0640#A

- Highlight the entry and press the right button of your mouse to get a menu. Select "delete entry" to remove the entry from CDS. - Select all applications, programs, and processes under ops and test mode that were registered and delete them all: /.:/ecs/GSFC/ops/xProgram_ops /.:/ecs/GSFC/ops/xProgram_ops_group /.:/ecs/GSFC/ops/xProgram_ops_profile - You may get out of cdsbrowser by select File, then Exit. Remember when logging out of DCE, type kdestroy.	

4.7.5.2 Test Case 2: External Interface Communication Test (BC006.002)

The External Interface Communication test represents an integration of the User Account Management test, the Kerberos ftp test, the Kerberos Client/Server Communication test, and the Security Authorization test.

The purpose of this test is to verify the capability to submit a request by a guest user for a registered user account, to notify the MSS Management Agent service by the CSS Security Service upon a predetermined number of unsuccessful login attempts, to send/receive both ASCII and binary files interactively and non-interactively between hosts using the file transfer protocol (FTP), and to send/receive files with kerberos encrypted data between local and remote terminals via kftp.

Test Configuration:

Hardware: MSS Server, SUN Solaris 2.4, SunOS 5.4, HPUX 9.05, and SGI:IRIX 5.3

Software: MsAcUsrProfile, MsAcUsrReqMgr, MsAcGuestUserReg, CsftFTPRelA.cxx, CsFtFTPRelA.h, ftptest.C, Kerberos COTs, Clientmain.cxx, ServerMain.cxx, CmdLineParser.cxx/.h, SleeperClient.cxx/.h, SleeperServer.cxx/.h, SleeperSecurity.cxx/.h

Data: Sample user information, ASCII/Binary files

Tools: DCE (rgy_edit), OODCE 1.0.3, RogueWave Tools.h++ 6.1.0 or version 5.17, Network analyzer: Sniffer, encryption tool, BuilderXcessory 3.5.1p

Test Input:

Inputs to this test include inserting user profile data into Sybase, EcSeLogin_3 executable, performing an ASCII or binary file transfer interactively to simulated local and remote DAAC. Perform a file transfer to SCF host (infrastructure) via ftp. Verify file transfers between the mini-DAAC and the infrastructure. Through the Kerberos application, a test data file is encrypted and transmitted across the FDDI network. The network analyzer attempts to access

the encrypted data file, reviewed its contents, and verify a transfer between the terminals in use. Transfer attempts are made via kftp.

Test Output:

Outputs to this test include new user accounts created and inserted into the database, notification to the MSS Management Agent upon a predetermined number of unsuccessful login attempts, connection to both local and remote DAAC hosts, complete file transfers interactively from the source to the destination without lost data. The network analyzer accesses the encrypted data file and displays a summary of data file.

Success Criteria:

This test is successful when the DCE account is created, the user appears in the MsAcUserProfile table in Sybase, a predetermined number of unsuccessful login attempts notify the MSS Management Agent service, files can be sent and received without errors, time/stamps, valid filesize match, checksums are valid, and files are not overwritten. The files are transferred between the local and remote hosts via kftp, and valid users obtain the information contained in the data files. The transferred data must be encrypted.

Test Procedures:

Test Case ID: BC006.002	
Test Name: External Interface Communication Test	
Test Steps:	Comments:
a. User Registration Request	
1. Login into msse4hp in the Release A mini-DAAC	
2. Set your terminal environment display enter: **setenv DISPLAY <IP address>**	This sets the environment display for the perspective machine/monitor where the user is logged.
3 From the command line on msse4hp, enter: **DCE_login cell_admin**	The cell_admin privilege is needed for DCE in order to create a keytab file for the executables.
4. Enter password for the cell_admin DCE account.	
5. Enter **rgy_edit <return>**	The rgy-edit is a cell_admin function that can only be initialized once the user is in DCE.
6. At the **rgy_edit** prompt, enter: >ktadd -p cell_admin -f MsAcUsrProfileServer. keytab	
7. Enter password for cell_admin and quit.	
8. Enter **kdestroy** while in the DCE mode.	The kdestroy command needs to be initiated before exiting DCE because it destroys a principal's login context and associated credentials that are created with each DCE session.
9. Enter exit to end the DCE session.	

10.From the msse4hp command line, enter: **cd /usr/testa/bin/ACC**	This directory path has the executables for User Registration.
11.From the msse4hp command line, enter: **runserver.csh**	Enables the Profile Server.
12.The following messages will appear: **Server is running, DB is ready** **Registered User server is ready**	
13.From the msse4hp command line, enter: **ps**	This displays the status for the MsAcServer. Please note the PID Id because it will be needed in step#16.
14.From the msse4hp command line, enter: **runclient.csh**	This executable will create, register, add user <username> into Sybase.
15.The runclient.csh driver will do the following: **Insert Request for <username>** **Retrieve Request List for <username>** **Create Registered User for <username>** **List Registered User for <username>** **List User Profile for for <username>** **List Daily, Weekly, Monthly Usage for <username>**	Gives User Profile information. C-MSS-75020
16.From the msse4hp command line, enter: **kill -9 <process #>**	This terminates the MsAcServer.
b. CSS To MSS Login Notification	
17. Enter: **cd /usr/testa/IT/tools/CSS/SEC**	change directory
18. Enter: **klist**	Verify credentials ticket not created
19. Enter: **EcSeLogin_3**	Display Usage instructions
20. Enter: **EcSeLogin_3 <username>**	Enter Valid Username
21. Enter Password: **<password>**	Enter Valid password
22. Enter: klist	Verify credentials ticket created
23. Enter: kdestroy	Destroy Ticket
24. Enter: **EcSeLogin_3 <username>**	Enter Valid Username
25. Enter Password: **<password>**	Enter Invalid Password; Repeat 2 times. Only 3 attempts are allowed.
26. Verify **Exit** from Password prompt.	
27. Enter: **klist**	Verify credentials ticket not created
28. Verify DCE login attempts notify MSS Agent.	
29. Enter: **EcSeLogin_3 <username>**	Enter Invalid Username.
30. Verify Invalid username rejected by DCE	ERROR: User ID Failure: Registry Object Not found (DCE/sec)
31. Enter: **EcSeLogin_3 <username> <password>**	Enter Valid username and Valid password on one line.
32. Enter: **klist**	Verify credentials ticket created
33. Enter: kdestroy	Destroy Ticket
34. Enter: **EcSeLogin_3 <username> <password>**	Enter Valid username and Invalid password on one line
35. Enter: **klist**	Verify credentials ticket created
36. Enter: **kdestroy**	Destroy Ticket
37. Verify DCE login attempts notify MSS Agent.	
c. File Transfer Protocol (FTP)	
38. Login to local DAAC host	
39. Type: **setenv DISPLAY 'host IP address':0.0** ex. setenv DISPLAY 155.157.123.25:0.0	Setting visualization for the user host
40. Set view in clearcase: cleartool setview 'username'	View is set for 'username'

41. Perform a ftp command to a local host. Enter: **ftp** *hostname*. Provide a user name and a password.	If an authorized user name and password are valid, then successful permission to enter the file transfer program is generated. Hence: the "*open"* command could also be used for ftp connections.
42. Verify the file transfer efficiency statistics is "on" Type: **verbose;** type again to turn "on" if status says "off "	Default is "on" Provides authentication to file transfer efficiency statistics locally and remotely
43. Check the local status: Type: **status**	Identifies the *local* connection host, mode, verbose status, store and receive status, case status, number of transmission status, number of default file name mappings, printing status, proxy connection if any, and case status
44. To ensure that the user does not overwrite a file (s) locally or remotely, check the current settings: (step 6) Type **runique**: for files being received Type **sunique**: for files being sent	no overwritten files via ftp (sent or received) Default is "off "
45. Type **status**	Verify the status is "on" for both runique and sunique
46. Select the file(s) to transferred locally **get** *filename* (one) **mget** *filename* (multiple)	Default mode is ASCII File (s) are transferred to the current directory
47. View output listing: **ls -al**	A listing of new file(s) transferred.
48. Create an error scenario: cannot capture a file (s)	
49. Change transfer mode to binary: Type: **binary**	Response: Type set to I
50. Repeat step 9	
51. Type: **ls -al**	verification of binary transferred file(s)
52. Set mode to default (ASCII) Type: ascii	Type set to A
53. Select the file(s) to be received locally **put** or **send** *filename* (one) **mput** *filename* (multiple)	File (s) from the user's current directory is sent to a local host
54. Type: **bye**	Exits ftp. Places user at its initial state
55. Type: **ls -al**	The initial state should show a received file in its current directory
56. Create an error scenario: c cannot retrieve a file (s)	
57. Re-enter *ftp*: Step 4	
58. Change transfer mode to binary: Type: **binary**	
59. Repeat step 16	Binary files are sent
60. Type: **bye**	Exits ftp. Places user at its initial state
61. Type: **ls -al**	verification of binary transferred file(s)
62. Again, Re-enter *ftp*: Step 4	
63. Transfer mode is (ASCII)	Default: ASCII
64. Initiate a close: **close**	The ftp session is closed.
65. 'Open' a session: **open hostname;** provide user login name and password	Independent ftp client process is started. (Interactive mode gives immediate responses)

66. Perform steps 5-14	Completed file transfer and verification
67. Type: **close**	closes a ftp client session
68. Type: **bye**	Exits ftp
69. Intermittedly verify file transfer type: Enter: type	Response: Using " "mode to transfer files
70. Repeat scenario twice more	
d. Kerberos File Transfer Protocol Test	
71. Login to local host	
72. Type: **setenv DISPLAY 'host IP address':0.0** ex. Setenv DISPLAY 155.157.123.25:0.0	Setting visualization for the user host
73. Setview in clearcase: cleartool setview 'username'	View is set for 'username'
74. Login to the kerberos COTS	
75. Locate any data file within kerberos and change format to encryption [done by the encryption tool]	
76. Begin data capture with the network analyzer.	
77. Initiate the cable tester and filtering	
On the workstation, perform a file transfer	
78. Perform a ftp command to a local host. Enter: **kftp** *hostname*. Provide a user name and a password.	If an authorized user name and password are valid, then successful permission to enter the file transfer program
79. Verify the file transfer efficiency statistics is "on" Type: **verbose;** type again to turn "on" if status says "off "	Default is "on" Provides authentication to file transfer efficiency statistics locally and remotely
80. Check the local status: Type: **status**	
81. To ensure that the user does not overwrite a file (s) locally or remotely, check the current settings: (step4) Type **runique**: for files being received Type **sunique**: for files being sent	no overwritten files via ftp (sent or received)
82.Type: **status**	Verify the status is "on" for both runique and sunique
83. Select the file(s) to transfer locally **get** *filename* (one) **mget** *filename* (multiple)	File (s) are transferred to the current directory
84. View output listing: **ls -al**	A listing of new file(s) transferred
85. Create an error scenario	
86. Type: **close**	closes ftp connection
87. Stop capturing data via the network analyzer: F10 option or ESC	
88. Using the network analyzer, ensure that all the files transferred via kftp are encrypted. Save to disk.	
89. Login to another host (valid user), and attempt to access the file	Only those with authorization will be able to access the file
90. Redo: Steps 74-87	
91. Repeat scenarios with invalid users	No authentication EOSD0500#A

4.8 Management Services 1 Tests

Management Services 1 tests consist of the following builds and threads:

- Performance Management Thread (TC020)
- Hardware Fault Management Thread (TC021)
- Management Data Access I Thread (TC022)
- Accountability Thread (TC023)
- Change Request Manager Thread (TC024)
- Management Services 1 Build (BC007)

4.8.1 Performance Management Thread (TC020)

The purpose of these tests is to verify the ability of the MSS Performance Management Application Service to provide performance data monitoring, trending, testing, and reporting. The Tivoli Management Environment is the chosen performance management tool for Release A. Along with the Tivoli application, HP OpenView, Management Data Access, Management Agent Services and OA Tools provide functionality required for performance management.

4.8.1.1 Test Case 1: Performance Monitoring of ECS Components (TC020.001)

This test case demonstrates the ability to monitor, display, and print the performance parameters of external and internal (site) network components on demand and at configured intervals. The test also verifies that the MSS performance management application service can receive requested/unrequested performance data from network components and ECS managed objects, and can determine the operational state of all network components, hosts, and peripherals that are either on-line, off-line, or in test mode.

Test Configuration:

Hardware: MSS Server, SNMP manageable routers, bridges, gateways and links.

Software: HP OpenView, Management Data Access (MDA), MACI

Data: N/A

Tools: N/A

Test Input:

Inputs to this test case include Tivoli and HpOpenView commands, Normal network traffic will be present on the system.

Test Output:

The expected outputs of this test include the ability to accurately and promptly report performance information for monitored network components.

Success Criteria:

All of the designated network components report the requested performance information in a timely manner.

Test Procedures:

Test Case ID: TC020.001	
Test Name: Performance Monitoring of Network Components	
Test Steps:	Comments:
1. Log onto MSS Server.	If remotely logged in, set display variable to reflect current console.
2. Initialize HP OpenView by entering: **ovw &**	
3. Click on IP Internet submap.	
4. Select appropriate submap to display desired network component.	
5. Verify the color of the component icon indicates correct operational state.	C-MSS-66120, C-MSS-68000
6. Highlight desired component on submap.	
7. Display event window for component.	Not available for links.
8. Verify performance related event information is shown.	Unrequested performance data (non-polled information) will be transmitted via the ECS event logging mechanism. C-MSS-66070
9. From ToolBar select the **Performance->Network Activity->Interface Statistics** menu.	This will cause the component to be polled for the desired MIB variables.
10. Verify the ability to display, and print the following: operational status (on-line, off-line, or test mode) type speed octets in/out packets in/out discards in/out errors in/out	C-MSS-66060, C-MSS-66050, C-MSS-68020
11. Repeat steps 4-7 for specified local and external routers, bridges, and gateways.	Steps 4-5 should be done to verify the operational state of hosts, peripherals, and links.
12. Manually change operational state of network component.	If shutdown of component is not feasible then, depending on its current state, place it in on-line or test mode.
13. Verify the color of the component's icon changes to reflect the change in operational state.	C-MSS-66130

14. Return component to prior operational state.	
15. Verify the color of the component's icon changes to reflect the change in operational state.	C-MSS-66130
16. Exit HP OpenView.	

4.8.1.2 Test Case 2: Performance Monitoring of Hosts (TC020.002)

This test case demonstrates the ability to retrieve, monitor, display, and print performance parameters and data for all hosts their applications on demand and at configured intervals.

Test Configuration:

Hardware: MSS Server, SNMP manageable hosts, routers, bridges, gateways and links.

Software: Tivoli, HP OpenView, Management Data Access (MDA), MACI

Data: N/A

Tools: N/A

Test Input:

Tivoli, and HpOpenView Network Node Manager commands.

Test Output:

The ability to accurately and promptly report performance information for monitored hosts and applications.

Success Criteria:

All of the designated hosts report the requested performance information within a specified minimum time frame.

Test Procedures:

Test Case ID: TC020.002	
Test Name: Performance Monitoring of Hosts	
Test Steps:	Comments:
1. Log onto MSS Server.	If remotely logged in, set display variable to reflect current console.
2. Initialize HP OpenView by entering: **ovw &**	
3. Select **Performance->Tivoli** menu	This will launch the Tivoli desktop.
4. Double click policy region containing desired hosts.	
5. Double click desired host icon.	
6. Double click performance profile.	

7. Verify the ability to display and print the following information: • total CPU utilization • memory utilization • physical disk I/Os • disk storage size • disk storage used • number of active processes • length of run queue • network I/0s (packets) • network errors	C-MSS-66100
8. Exit Tivoli.	
9. In HP OpenView, select submap containing desired host.	
10. Open map for desired host.	
11. Select application shown on host map.	
12. From toolbar select the Performance->Application Statistics menu.	
13. Verify the ability to display and print the following information: • algorithm name (if applicable) • algorithm version (if applicable) • start time • stop time • CPU utilization • memory utilization • disk reads • disk writes	C-MSS-66305, C-MSS-66310
14. Exit HP OpenView	

4.8.1.3 Test Case 3: Performance Monitoring of Communication Protocol Stacks (TC020.003)

This test case demonstrates the ability to collect, monitor, display, and print the performance information of communication protocol stacks on managed devices.

Test Configuration:

Hardware: MSS Server, SNMP manageable hosts, routers, bridges, gateways and links.

Software: Tivoli, HP OpenView, Management Data Access (MDA), MACI

Data: N/A

Tools: N/A

Test Input:

Tivoli, and HpOpenView Network Node Manager commands.

Test Output:

The expected results of this test include the ability to accurately and promptly report performance information for communication protocol stacks on managed devices.

Success Criteria:

This test is successful when all of the designated communication protocol stacks report the requested performance information in a timely manner.

Test Procedures:

Test Case ID: TC020.003	
Test Name: Performance Monitoring of Communication Protocol Stacks	
Test Steps:	Comments:
1. Log onto MSS Server.	If remotely logged in, set display variable to reflect current console.
2. Initialize HP OpenView by entering: **ovw** &	
3. Select **Performance->Tivoli** menu	This will launch the Tivoli desktop.
4. Double click policy region containing desired hosts.	
5. Double click desired host icon.	
6. Double click performance profile.	
7. Verify the ability to display, and print the following information: • number of transport layer messages received with errors • number of transport layer messages requiring retransmission • number of transport layer messages received that could not be delivered • number of network layer messages received with errors • number of network layer messages received that could not be delivered • number of network layer messages that were discarded	C-MSS-66090
8. Exit Tivoli desktop.	
9. Click on IP Internet submap.	
10. Select appropriate submap to display desired network component.	
11. Highlight desired component on submap.	
12. From ToolBar select the **Performance->Network Activity->Interface Errors** menu.	
13. Verify the ability to display, and print the following information: • number of transport layer messages received with errors • number of transport layer messages requiring retransmission • number of transport layer messages received that could not be delivered • number of network layer messages received with errors • number of network layer messages received that could not be delivered • number of network layer messages that were discarded	C-MSS-66090
14. From ToolBar select the **Misc->SNMP Browser** menu	

15. Query the **iso.org.dod.internet.mgmt.mib-2** tree to verify the monitoring of RFC 1213 and RFC 1623 performance parameters.	C-MSS-66020, C-MSS-66030
16. Select a MIB at the end of the mib-2 tree.	
17. Click on the **Describe** button in the MIB browser.	
18. Verify the MIB definitions is shown.	C-MSS-66030
19. Repeat steps 10-15 for other network components.	
20. Exit HP OpenView.	

4.8.1.4 Test Case 4: Performance Monitoring Thresholds (TC020.004)

This test case demonstrates the ability of the MSS Performance Management Application Service to provide the following:

- a configurable number of thresholds for each performance metric.
- allow each performance metric threshold to be configurable.
- compare each received performance metric value against these defined thresholds.
- extract values of performance metrics gathered for a specified managed object over a configurable period of time from the Management Database.

Test Configuration:

Hardware: MSS Server, SNMP manageable hosts, routers, bridges, gateways links.

Software: Tivoli, HP OpenView, Management Data Access (MDA), MACI

Data: N/A

Tools: N/A

Test Input:

Tivoli, and HpOpenView Network Node Manager commands.

Test Input:

Inputs to this test case include HpOpenView Network Node Manager commands and ECS performance data.

Test Output:

The expected outputs of this test include the ability to accurately compare received performance metrics to configured thresholds.

Success Criteria:

This test is successful when all performance parameters exceeding their configured thresholds are flagged.

Test Case ID: TC020.004	
Test Name: Performance Monitoring Thresholds	
Test Steps:	Comments:
1. Log onto MSS Server.	If remotely logged in, set display variable to reflect current console.
2. Initialize HP OpenView by entering: **ovw** &	
3. Select **Performance->Tivoli** menu	This will launch the Tivoli desktop.
4. Double click policy region containing desired hosts.	
5. Double click desired host icon.	
6. Double click performance profile.	
7. Verify default thresholds are displayed for all performance metrics.	C-MSS-66190, C-MSS-66200, C-MSS-66210, C-MSS-66220
8. Select the metrics to be monitored.	C-MSS-66040
9. Change the threshold for a given performance metric so that it will trigger on next polling interval.	C-MSS-66230
10. Verify that appropriate indicator icon signals threshold has been crossed.	C-MSS-66240
11. Verify message appears in notices group indicating threshold has been crossed.	C-MSS-66250, C-MSS-66170
12. Return threshold to prior value.	
13. Exit Tivoli desktop.	
14. In HP OpenView window, click on IP Internet submap.	
15. Select appropriate submap to display desired network component.	
16. Highlight desired component on submap.	
17. From ToolBar select the **Options->Data Collections and Thresholds** menu.	
18. Verify a list of performance metrics are show with their associated default thresholds.	C-MSS-66190, C-MSS-66200
19. Select the metrics to be monitored.	C-MSS-66040
20. Change the threshold for a given performance metric so that it will trigger on next polling interval.	C-MSS-66230
21. Verify that on the next polling interval Event Window indicates the threshold being crossed.	C-MSS-66240, C-MSS-66250, C-MSS-66170
22. Click on the **Actions->Show** Data menu to display previously collected data for the selected performance metrics.	C-MSS-67000
23. Return threshold to previous value.	
24. Repeat steps 14-21 for other network components.	
25. Exit HP OpenView.	

4.8.1.5 Test Case 5: History Log Verification (TC020.005)

This test case demonstrates the ability of the MSS Performance Management Application Service to:

- request and receive performance and summarized performance data from site performance management applications and other external systems.
- generate statistics for a configurable period of time for performance data stored in the Management Database.

- record an event in the local History Log.
- provide queries that generate performance statistics.
- store generated performance statistics.
- extract and send summarized site status information to the MSS EMC performance management application service which logs the received data.

Test Configuration:

Hardware: MSS Server, MSS SMC Server SNMP manageable hosts, routers, bridges, gateways links.

Software: Tivoli, HP OpenView, Management Data Access (MDA), MACI

Data: N/A

Tools: N/A

Test Input:

Inputs to this test case include HpOpenView Network Node Manager commands, APIs, and ECS performance data.

Test Output:

The expected outputs of this test include the ability to write and retrieve all selected performance management data to and from the History Log. Also, performance and summarized performance data from the performance data at the local sites.

Success Criteria:

This test is successful when all selected performance management data has been properly written to and retrieved from the History Log, after which statistics are extracted, summarized and sent to the SMC.

Test Procedures:

Test Case ID: TC020.005	
Test Name: History Log Verification	
Test Steps:	Comments:
1. Examine the History Log to ensure that all performance parameters which exceeded their configured thresholds have been recorded.	
2. Log onto MSS Server.	If remotely logged in, set display variable to reflect current console.
3. Initialize HP OpenView by entering: **ovw** &	
4. Select **Performance->Tivoli** menu	This will launch the Tivoli desktop.

5. Using Tivoli commands, extract summarized site information from logged performance data via query tool containing the following information the requested performance metrics: • average • median • maximum • minimum • ratios • rates • standard deviations	C-MSS-66140, C-MSS-66150, C-MSS-66160, C-MSS-66180, C-MSS-66260, C-MSS-66280
6. Save the generated statistics to a file.	C-MSS-66270
7. Verify that the site can send summarize data to the EMC Tivoli Desktop.	C-MSS-66290
8. Verify that the EMC Tivoli Desktop can request and receive performance and summarized performance data from the site PMAS and other external systems.	C-MSS-66140, C-MSS-66150, C-MSS-66160
9. At the EMC, store the summarized size reports	C-MSS-66300
10. Exit Tivoli and HP OpenView.	

4.8.1.6 Test Case 6: Performance Trending (TC020.006)

This test case demonstrates the ability of the MSS Performance Management Application Service to graph the extracted performance metrics gathered for specified network objects.

Test Configuration:

Hardware: MSS Server, SNMP manageable hosts, routers, bridges, gateways links.

Software: HP OpenView, Management Data Access (MDA), MACI

Data: N/A

Tools: History Log populating scripts.

Test Input:

Inputs to this test case include HpOpenView Network Node Manager commands, ECS performance data stored in the History Log, and scripts to populate the History Log.

Test Output:

The expected outputs of this test include the ability to extract and graph all selected performance management data from the History Log for a selected time period.

Success Criteria:

This test is successful when all selected performance management data has been properly retrieved from the History Log and graphed for the selected time interval.

An error message should be displayed when a null interval is selected.

Test Procedures:

Test Case ID: TC020.006	
Test Name: Performance Trending	
Test Steps:	Comments:
1. Use History Log populating scripts to enter additional performance data covering approximately a two-year period.	
2. Log onto MSS Server.	If remotely logged in, set display variable to reflect current console.
3. Initialize HP OpenView by entering: **ovw** &	
4. Using HpOpenView Network Node Manager commands, extract performance metrics covering a two-year period.	
5. Using HpOpenView Network Node Manager commands, generate a graph of the performance metrics extracted above.	C-MSS-67010
6. Repeat the above two steps for a five-minute interval.	
7. Exit HP OpenView.	

4.8.1.7 Test Case 7: Performance Benchmark Testing (TC020.07)

This test case demonstrates the ability of the MSS Performance Management Application Service to receive, maintain, and perform operational network performance benchmark test procedures.

Test Configuration:

Hardware: MSS Server, SNMP manageable hosts, routers, bridges, gateways links.

Software: HP OpenView, Management Data Access (MDA), MACI,

Data: operational benchmark procedures

Tools: OA Tools

Test Input:

Inputs to this test case include HpOpenView Network Node Manager commands, and operational benchmark test procedures

Test Output:

The expected results of this test include the operational network performance benchmark measurements and test results.

Test Output:

This test is successful when all operational network performance benchmark tests pass under normal network conditions. If any error or warning conditions exist on the network, they should be flagged.

Test Procedures:

Test Case ID: TC020.007	
Test Name: Performance Benchmark Testing	
Test Steps:	Comments:
1. Ensure that the MSS Performance Management Application Service operational network performance benchmark test procedures reside in their designated location.	Maintained via a Word Processor. C-MSS-69000
2. Perform all operational network performance benchmark test procedures.	C-MSS-69020
3. Examine test results for any discrepancies.	
4. Repeat all tests several times, and compare the results.	
5. Introduce error conditions into the network, then re-run the operational network performance benchmark test procedures.	
6. Generate report from collected results using OA Tools..	C-MSS-68090
7. Save file to a disk and then print report.	C-MSS-68100
8. Forward report to the Performance Management Application Service for M&O verification and validation.	C-MSS-69010, C-MSS-69020

4.8.1.8 Test Case 8: SDPS Subsystems Interfaces (TC020.008)

This test case demonstrates the ability of the MSS Performance Management Application to receive and provide system resource utilization information requests from/to the SDPS Data Processing subsystem, Data Server subsystem, and Client subsystem.

Test Configuration:

Hardware: MSS Server, Processing Server, Data Server, Client Workstation

Software: HP OpenView, MDA, MACI, SDPS Data Processing, Data Server, and Client subsystem.

Data: respective subsystems' resource utilization information

Tools: N/A

Test Input:

Inputs to this test case include Tivoli and HP OpenView commands, and resource utilization information.

Test Output:

Outputs include performance statistics and current resource utilization information. Also, reports of management data in a variety of outputs.

Success Criteria:

This test is successful when the resource utilization information is provided by the MSS performance management application service and the SDPS, Data Server, and Client subsystems can receive it.

Test Procedures:

Test Case ID: TC020.0008	
Test Name: SDPS Subsystem Interfaces	
Test Steps:	Comments:
1. Log onto MSS Server.	If remotely logged in, set display variable to reflect current console.
2. Initialize HP OpenView by entering: **ovw** &	
3. Select **Performance->Tivoli** menu	This will launch the Tivoli desktop.
4. Verify that Tivoli is capable of receiving system resource utilization information requests from SDPS Data Processing Subsystem, SDPS Data Server and Data Processing Subsystems and SDPS Client subsystem.	C-MSS-68030, C-MSS-68050, C-MSS-68070
5. Verify that the Tivoli is capable of providing CPU utilization, memory utilization and disk I/O's to the SDPS Data Processing Subsystem, SDPS Data Server and Data Processing subsystems, and SDPS Client subsystem.	C-MSS-68040, C-MSS-68060, C-MSS-68080
6. Exit Tivoli and HP OpenView.	

4.8.2 Hardware Fault Management Thread (TC021)

Hardware Fault Management addresses the detection, isolation, diagnosis, and recovery from a fault condition in a managed hardware object, and the restoration of the affected system or service to an operational state. The managed hardware objects in Release A for which Fault Management is provided include network devices, operating systems, and peripheral devices. These managed objects include those of SDPS and CSMS.

The Hardware Fault Management Application Service (FMAS) comprises the following functional services:

- Fault Definition and Setup (TC021.001)
- Fault Detection and Notification (TC021.002)
- Fault Diagnosis, Isolation and Identification (TC021.003)
- Fault Policies and Procedures (TC021.004)
- Fault Recovery (TC021.005)

- Reporting (TC021.006)

The objective of this thread is to verify the above mentioned functional services by demonstrating each of the service requirements successfully.

4.8.2.1 Test Case 1: Fault Definition and Setup (TC021.001)

The purpose of the Fault Definition and Setup test is to demonstrate the management framework's ability to create and display graphical representations of network topologies. Also tested is the ability to organize the given network topology into a hierarchy of maps. The FMAS will be tested to demonstrate the following functions:

- define fault categories
- assign faults to categories
- assign severity levels to faults

Testing demonstrates the FMAS capability to provide the Management Data Access Service with a configurable list of fault categories that specify whether to enable or disable the logging of fault notifications for that fault category. Testing also demonstrates the capability to specify additional information to be added to a disk log file, based on the fault category, when the notification of a fault is received.

Test Configuration:

Hardware: HP 9000/770

Software: HP-UX 9.05, HP OpenView B.04.00

Data: Managed hardware/software objects

Tools: DCE, Network Analyzer/Sniffer

Test Input:

Test inputs include the SNMP managed objects that display the following graphical representations of a given network topology:

- routers
- communication lines
- hosts
- peripherals
- applications

Test Output:

The expected outputs of this test include an accurate map display of the operational network with lower level maps including individual sites, workstations, and network topologies. Also, a successful demonstration of the fault management functions described in the test inputs.

Success Criteria:

This test is successful when maps reflecting the network topology are displayed and the fault management requirements are successfully demonstrated.

Test Procedures:

Test Case ID: TC021.001	
Test Name: Fault Definition and Setup	
Test Steps:	Comments:
The following steps verify that the SNMP protocol is being used:	Configuration and control is also displayed throughout this test case: C-MSS-16030 (P2 & P3)
1. Set up the Network Analyzer/Sniffer to monitor all traffic with the machine on which HP OpenView resides, ensuring that the filter is set to capture only SNMP traffic.	SNMP traffic should be captured. C-MSS-16005
The following steps verify the graphic topology:	
2. Log onto the designated FMAS machine. If remotely logged in, set the DISPLAY variable properly.	
3. Initialize HP OpenView by entering: **ovw &**	HP OpenView B.04.00 should come up and run in the background. C-MSS-12010 EOSD1703#A
4. Verify that a map depiction of the network topology is accurately displayed.	C-MSS-12005 C-MSS-12020 C-MSS-14010 C-MSS-60010
5. Verify that the lower level topologies include hosts, routers, communication lines, peripherals, and applications.	P3 - applications
6. Verify their status condition.	
7. Double-click on the available icons to verify that the lower level submaps exist and are accurately displayed.	C-MSS-14030 C-MSS-20010
The following steps verify symbol manipulation:	
8. Using the “Edit” option, add a symbol to the map currently displayed.	C-MSS-12040 C-MSS-14020
9. Using the “Edit” option, delete a different symbol from the map currently displayed.	C-MSS-12040 C-MSS-14020
10. Click on a symbol on the map with the middle mouse button to drag it to a different location.	C-MSS-12040
11. Click on a symbol on the map with the right mouse button, then select the “Describe/Modify Symbol” option from the resulting pull-down menu.	The Symbol Description window will be displayed. C-MSS-12090
12. In the Symbol Description window, ensure that the “Status Source” is set to “Object”.	This step is necessary to enable a change in symbol status (i.e., color). C-MSS-12140
13. From the Unix window where HP OpenView was initialized, enter the following for the object in Step 12: **/usr/OV/contrib/NNM/setStatus/setStatus** *<object>* *<new_status>*	The status/color of the object should change to the new status/color. C-MSS-12040

14. Click on a symbol on the map with the right mouse button, then select the “Describe/Modify Symbol” option from the resulting pull-down menu.	The Symbol Description window will be displayed.
15. In the Symbol Description window, ensure that the “Status Source” is set to “Symbol”.	This step is necessary to enable a change in symbol type (i.e., shape).
16. Click on the symbol from Step 14 with the right mouse button, then select the “Change Symbol Type” option from the resulting pull-down menu.	The Change Symbol Type window will be displayed.
17. In the Change Symbol Type window, select a new symbol class and subclass, then click on the OK button.	The symbol type/shape of the object should change to the new type/shape. C-MSS-12040
18. Highlight a managed symbol, then select Unmanage Objects from the Map pull-down menu.	The symbol should turn grey, and a change may occur in the state of any connected objects. C-MSS-12170 C-MSS-20040
19. Highlight the unmanaged symbol from above, then select Manage Objects from the Map pull-down menu.	The symbol and connected objects should return to their previous states. C-MSS-12170 C-MSS-20040
The following steps verify the fault categories:	
20. Define a fault category via the Data Collection & Thresholds: SNMP window (Event Configurator sub-window).	C-MSS-60020
21. Assign specific faults to the defined category via the Data Collection & Thresholds: SNMP window (Event Configurator sub-window).	C-MSS-60030
22. Assign severity levels to each fault via the Data Collection & Thresholds: SNMP window (Event Configurator sub-window).	C-MSS-60040
The following steps verify the enabling/disabling of fault notifications:	
23. Bring up the Event Configurator window via the Collection & Thresholds: SNMP window.	
24. Select the “Don’t log or display” option from the Event Category section.	C-MSS-60050 ESN-0910#A
25. Force the selected event to occur.	Ensure that it has not been logged.
26. Return to the Event Configurator window, and select “Log only” from the Event Category section.	
27. Force the selected event to occur.	Ensure that it has been logged.
The following steps verify the capability of defining polling frequency.	
28. Modify the current polling rate for a MIB object via the Data Collection and Thresholds window.	Ensure that the polling rate has changed. C-MSS-60200
The following steps verify the display of on-line help windows:	
29. Examine the upper, right hand corner of all major HP OpenView windows.	Ensure that a Help pull-down menu label exists.
30. Select several menu items from the Help pull-down menu.	Ensure that the resultant Help windows contain information pertinent to its parent window. C-MSS-12180

4.8.2.2 Test Case 2: Fault Detection and Notification (TC021.002)

The purpose of this test case is to verify the ability to detect faults via two methods: 1) by polling an attribute of a managed object, and 2) by receiving notification from another entity. Fault detection may be accomplished by the FMAS itself for network devices, or by an agent for defined managed objects, such as processes, databases, peripheral devices, and applications.

The FMAS will be tested to demonstrate several mechanisms for fault notification. These include visual indications/notifications, such as changing an icon color or displaying a message in a pop-up notification window, logging the notification to a disk log file, and generating audible alerts.

The FMAS generates faults when a defined threshold is exceeded and/or when notification is received from SNMP agents and applications. All fault notifications are logged for record keeping, report generation and post processing.

FMAS will be tested to demonstrate the capability to generate the following types of notifications for detected faults:

a. Color change for icon display

b. Message in a pop-up notification window

c. Disk file logging the following fault information

   1. fault type
   2. date/time of fault occurrence
   3. source IP address of the notification
   4. fault data received with notification
   5. operator-defined descriptive text

d. Audible alert

Also tested is the ability to generate a fault notification within five minutes of fault detection.

<u>Test Configuration:</u>

Hardware: HP 9000/770

Software: HP-UX 9.05, HP OpenView B.04.00

Data: Managed hardware/software objects

Tools: DCE, Network Analyzer/Sniffer

Test Input:

The test inputs include demonstrating that the FMAS has the capability to poll for the detection of fault and performance information, by polling network elements at a frequency that is defined in the system and receiving fault notifications from those elements.

The following types of failures will be tested for proper detection by the FMAS:

a. communication software version mismatch errors

b. communication software configuration errors

c. communication hardware errors

   1. unreachable host

   2. unreachable router

   3. communication link failures/errors

d. protocol errors

e. peripheral errors

f. database errors

g. application errors

   1. missing process (Application or COTS product)

   2. process in loop

Test Output:

The expected outputs of this test include a successful demonstration of the fault management functions described in the test inputs.

Success Criteria:

This test is successful when the fault management requirements are successfully demonstrated.

Test Procedures:

Test Case ID: TC021.002	
Test Name: Fault Detection and Notification	
Test Steps:	Comments:
The following steps verify that the SNMP protocol is being used:	
1. Set up the Network Analyzer/Sniffer to monitor all traffic with the machine on which HP OpenView resides, ensuring that the filter is set to capture only SNMP traffic.	SNMP traffic should be captured. C-MSS-16005
Initiate HP OpenView:	

2. Log onto the designated FMAS machine. If remotely logged in, set the DISPLAY variable properly.	
3. Initialize HP OpenView by entering: **ovw &**	HP OpenView B.04.00 should come up and run in the background. C-MSS-12010 EOSD1703#A
4. Verify that a map depiction of the network topology is accurately displayed.	C-MSS-12005 C-MSS-12020
The following steps verify the detection and notification types of various failure types.	P2 & P3: C-MSS-60150 C-MSS-60160 C-MSS-60170 C-MSS-60180 C-MSS-60190
5. Disengage network element (cause failure).	
6. Verify receipt of element failure detection from network display/alarm.	C-MSS-12090 C-MSS-16070 C-MSS-20020 C-MSS-20030
7. Verify correct failure detected.	Failure should be propagated up the hierarchical tree. C-MSS-14040 C-MSS-60340 ESN-0780#A ESN-0810#A SMC-3390#A SMC-3395#A
8. Verify fault detected within time frequency set in system.	C-MSS-60100
9. Verify proper color change for icon displayed.	C-MSS-60110 C-MSS-60200
10. Verify message pop-up notification window received.(when configured via optional field in Event Configurator window).	C-MSS-16050 (P2 & P3) C-MSS-60110 C-MSS-60200
11. Verify audible alert received (when configured via optional field in Event Configurator window).	C-MSS-16050 (P2 & P3) C-MSS-60110 C-MSS-60200
12. Insert additional information concerning the fault into the disk log file.	P3: C-MSS-60070
13. Repeat for all element failure types.	
The following steps verify fault support registration list:	
14. Create a fault distribution list of support providers.	M&O staff
15. Generate fault notification by network element failure.	
16. Verify correct failure notification received by all registered recipients within a five minute time period.	P3: C-MSS-12080 C-MSS-60220 C-MSS-60230

4.8.2.3 Test Case 3: Fault Diagnosis, Isolation, and Identification (TC021.003)

The purpose of this test case is to demonstrate that the Fault Management Application Service provides the required mechanisms to diagnose, isolate, and identify faults. These mechanisms include diagnostic tests, vendor-provided diagnostics, and disk log files containing diagnostic information. The FMAS debugging aid is tested to verify that it is capable of:

a. identifying routes between selected pairs of hosts

b. testing the:

1. connectivity of pairs of ECS hosts and routers
2. ability to reach hosts and routers
3. availability of network services

c. periodic testing of all ECS communication links to verify that they ar operational

d. verifying operational status of a host

e. periodic diagnostic testing for isolation and identification of faults

f. executing vendor diagnostics

g. providing isolation, location, identification, and characterization of fault data

Test Configuration:

Hardware: HP 9000/770

Software: HP-UX 9.05, HP OpenView B.04.00

Data: Managed hardware/software objects

Tools: DCE, Network Analyzer/Sniffer

Test Input:

Test inputs include operator actions perform diagnostics and testing for the following isolated faults:

a. connectivity between ECS hosts and ECS routers

b. ability to reach hosts and routers

c. availability of network services at hosts

Test Output:

The expected results of this test include a successful demonstration of the fault management functions described in the test inputs.

Success Criteria

This test is successful when the fault management requirements are successfully demonstrated.

Test Procedures:

Test Case ID: TC021.003	
Test Name: Fault Diagnosis, Isolation and Identification	
Test Steps:	Comments:
The following steps verify that the SNMP protocol is being used:	
1. Set up the Network Analyzer/Sniffer to monitor all traffic with the machine on which HP OpenView resides, ensuring that the filter is set to capture only SNMP traffic.	SNMP traffic should be captured. C-MSS-16005
Initiate HP OpenView:	
2. Log onto the designated FMAS machine. If remotely logged in, set the DISPLAY variable properly.	
3. Initialize HP OpenView by entering: **ovw &**	HP OpenView B.04.00 should come up and run in the background. C-MSS-12010 EOSD1703#A
4. Verify that a map depiction of the network topology is accurately displayed.	C-MSS-12005 C-MSS-12020
The following steps verify fault isolation and identification capabilities:	P2 & P3: C-MSS-60370 C-MSS-60390
5. Via the Event Configurator window, modify the Command for Automatic Action field to enable the audible alarm feature for a specific event.	C-MSS-12090
6. Force the above event to occur.	An audible alarm should be generated. C-MSS-60200
7. Examine the Error Events Browser and Status Events Browser.	The fault should be identified in the log, stating the fault type, date/time of occurrence, fault source, and descriptive text. C-MSS-60200 C-MSS-60395 SMC-4310#A
8. Repeat steps 6 & 7 for other faults.	
9. Using the Event Browser window's View option, set filter option prevent the display of certain events.	Ensure that the filter option prevents the display of certain events in the Event Logs. C-MSS-60060
10. Using the Event Configurator window, set the Event Category to "Don't log or display".	Ensure that the chosen event is not displayed. C-MSS-60060
The following steps verify diagnostic capabilities:	C-MSS-60350
11. Select the Network Connectivity: Poll Mode option from the Fault pull-down window for a configured object.	The Poll Mode window should come up and display polling information, indicating connectivity status for that object. ESN-0810#A ESN-0920#A SMC-4311#A

12. Select the Ping option from the Fault pull-down window for a configured object.	The Ping window should come up and display the number of packets sent from the local management system to the selected object, along with the number of packets lost and round trip times. C-MSS-60330 ESN-0810#A ESN-0920#A SMC-4311#A
13. Select the Test IP/TCP/SNMP option from the Fault pull-down window for a configured object.	The Test IP/TCP/SNMP window should come up and display the results of connectivity testing for the IP, TCP, and SNMP network protocols between the local management system and the selected object. C-MSS-16005 C-MSS-16100 C-MSS-60320 ESN-0810#A ESN-0920#A SMC-4311#A
14. Set up trap to monitor MIB variables for the detection of UDP/ICMP protocol errors.	Verify correct monitoring via generation of protocol errors and also without errors. P3: C-MSS-16020 C-MSS-16040 C-MSS-16100 C-MSS-60320
15. Select the Locate Route: via SNMP option from the Fault pull-down window for two nodes.	The Locate Route window should come up and display the specific IP routes between two selected nodes. C-MSS-16005 C-MSS-60300 ESN-0810#A ESN-0920#A SMC-4311#A
16. Toggle an object between operational mode and test mode.	P3: C-MSS-16010
17. Perform vendor diagnostics on hardware equipment.	P3: C-MSS-60360

4.8.2.4 Test Case 4: Fault Policies and Procedures (TC021.004)

This test case demonstrates that the minimum policies and procedures will be capable of:

- receiving fault management policies and procedures from the EMC
- interfacing with the MSS CMAS and scheduling a change in the configuration to recover from a fault

Test Configuration:

Hardware: HP 9000/770

Software: HP-UX 9.05, HP OpenView B.04.00

Data: Managed hardware/software objects

Tools: DCE

Test Input:

Test input requires the FMAS to be able support, maintain, and update the system fault management policies and procedures which include:

- Fault Identification
- Fault priorities
- Recovery and corrective actions

Test Output:

The expected outputs of this test include a successful demonstration of the fault management functions described in the test inputs.

Success Criteria:

This test is successful when the fault management requirements are successfully demonstrated.

Test Procedures:

Test Case ID: TC021.004	
Test Name: Fault Policies and Procedures	
Test Steps:	Comments:
1. Verify the contents of the file containing the list of external service providers, M&O operators, and applications to be notified when a specified fault is detected.	C-MSS-60210
2 Verify the receipt of fault management policies and procedures from EMC.	C-MSS-60410
3. Verify the contents of the file containing system fault management policies and procedures for fault identification, fault priorities, and recovery/corrective actions.	C-MSS-60400
4. Verify procedures for recovery from conditions of performance degradation and faults with sites and external service providers.	C-MSS-60500
5. Verify sending/receipt of directives/instructions for site fault recovery.	C-MSS-60510

4.8.2.5 Test Case 5: Fault Recovery (TC021.005)

The purpose of this test case is to verify the recovery procedures that will be initiated in order to restore the system to an operational state. These recovery procedures may be simple (the resetting of file permissions and restarting an application that aborted due to an error accessing a file), or more involved (scheduling corrective maintenance of failed equipment). In the latter case, some coordination with the Configuration Management Application may be necessary. Once the fault condition has recovered, the failed component may be restored to an operational state. FMAS must demonstrate the capability to provide the specification and execution of action routines in response to the notification of a fault and be capable of passing parameters to the

action routines. Automation support tools will be used for the recovery from faults within the site. Specific capabilities tested include the ability to:

- allow the specification and execution of action routines in response to a fault.
- pass parameters to action routines.
- use office automation tools to generate directives and instructions for fault recovery.
- coordinate the recovery of performance degradation and faults with the site and external network providers.
- coordinate the recovery of faults reported from a site.

Test Configuration:

Hardware: HP 9000/770

Software: HP-UX 9.05, HP OpenView B.04.00

Data: Managed hardware/software objects

Tools: DCE, Office Automation Tools

Test Input:

Test inputs include verifying that the EMC FMAS coordinates the recovery from conditions of performance degradation and faults with the sites and external network service providers.

Test Output:

The expected outputs of this test include a successful demonstration of the fault management functions described in the test inputs.

Success Criteria:

This test is successful when the fault management requirements are successfully demonstrated.

Test Procedures:

Test Case ID: TC021.005	
Test Name: Fault Recovery	
Test Steps:	Comments:
1. Following the initiation of a fault condition, as per TC021.02, ensure that configured action routines are executed in response to the fault.	C-MSS-60520 C-MSS-60530
2. Ensure that office automation tools can be used to generate fault recovery directives for use within a site.	C-MSS-60540
3. Ensure that a configuration change can be scheduled when needed for site configuration changes.	C-MSS-60420

4.8.2.6 Test Case 6: Fault Reporting (TC021.006)

The purpose of this test case is to verify the capability of FMAS to gather fault data that can be reported to M&O staff and external service providers in a graphical and tabular format via the MUI. Fault Management Application Service must also demonstrate the capability to provide the M&O staff with the ability to select and generate fault statistics (real-time and historical) for operator-selectable managed objects. These statistics must be displaceable in either a tabular or a graphical format.

Test Configuration:

Hardware: HP 9000/770

Software: HP-UX 9.05, HP OpenView B.04.00

Data: Managed hardware/software objects

Tools: DCE

Test Input:

Test inputs require FMAS to generate, on an interactive and scheduled basis, reports on performance/error data, that it has been configured to collect along, with the capability to build histories for different types of errors and events.

Test Output:

The expected outputs of this test include a successful demonstration of the fault management functions described in the test inputs.

Success Criteria:

This test is successful when the fault management requirements are successfully demonstrated.

Test Procedures:

Test Case ID: TC021.006	
Test Name: Fault Reporting	
Test Steps:	Comments:
Initiate HP OpenView:	
1. Log onto the designated FMAS machine. If remotely logged in, set the DISPLAY variable properly.	
2. Initialize HP OpenView by entering: **ovw &**	HP OpenView B.04.00 should come up and run in the background. C-MSS-12010 EOSD1703#A
3. Verify that a map depiction of the network topology is accurately displayed.	C-MSS-12005 C-MSS-12020
The following steps verify fault reporting capabilities:	

4. Select the Graph SMP Data option from the Performance pull-down window.	The appropriate graph will be displayed in the Graph All Collected Data window. C-MSS-60600 (P2 & P3) C-MSS-60610 C-MSS-60620
5. Select the Print option from the File pull-down menu of the Graph All Collected Data window.	The Print Graph All Collected Data window will be displayed.
6. Enter the following print command in the Print Graph All Collected Data window: **cat>/tmp/<***file_name***> ; xpr -device ps /tmp/<***file_name***> | lp -d <***printer***>**	The displayed graph will be printed. C-MSS-60620
7. Generate and print a graph from the Browse MIB window.	See general steps/comments from steps 4 to 6 above.

4.8.3 Management Data Access I Thread (TC022)

The Management Data Access (MDA) service is responsible for centralizing, processing, and providing access to information logged in the management data log file on each managed host. The MSS Agent Services collect management information from carious sources and store it in the MDA log file. This log data includes performance, security, fault, accountability, and other ECS application event information.

This thread verifies the ability of the ECS Log Browser (ELB) to browse two types of logs on a host:

- The Management Log containing information on management events.
- The Application Log containing information that an ECS application records for its own purposes.

4.8.3.1 Test Case 1: ECS Log Browser for MSS (TC022.001)

This test case verifies the capability of the ECS Log Browser (ELB) to allow a user to browse the Management log file. The Management log file is created by the EcAgEvent Manager and the log file contains information on management events.

The capabilities tested include:

- providing a host, time period, and selection filter information
- allowing an operator to selectively read a record from a Management log file
- retrieving the requested data and displaying brief description events
- once displayed, options are given for sorting, additional filtering, editing, and displaying detail event fields

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MDA User Interface, MDA Services

Data: Management log file, XRunner script

Tools: MsMdPhase2 (test driver), XRunner 4.0

Test Input:

Inputs to this test case include keyins to open and read a record from a Management log file.

Test Output:

Outputs to this test case include the description of an event field and an XRunner script.

Success Criteria:

This test is successful when the Management Log containing information on management events can be browsed based on the selection filter information and the requested data is retrieved, displayed, and sorted. The successful creation and execution of an XRunner script is also required.

Test Procedures:

Test Case ID: TC022.001	
Test Name: ECS Log Browser for MSS	
Test Steps:	Comments:
1. On the test machine (SUN5), set the environment variable of FileDir with *log at the end: **setenv FileDir <***/usr/testa/kmc/*log***>**	This test cannot run on the HP machine in the mini-DAAC (msse4hp) with XRunner. At the present time, there isn't an HP license for XRunner.
2. Set the Display environment variable: **setenv DISPLAY** <*host ip address*>:0.0	
3.Change directory to where the test drivers reside: **cd** <*/usr/testa/kmc*>	
4. Execute the test program: **MsMdPhase2 &**	The MsMdPhase2 application will run in the background. The msse3sun terminal will be in the interactive mode. The ECS Log Browser is displayed on the screen.
5. From the command line: **ps**	This will verify that the MsMdPhase2 process is running.
6. Open another SUN5 session: **select xterm from the pulldown menu**	This session will be used for XRunner.
7. On the second SUN5 session (msse3sun): **setenv DISPLAY** <*host ip address*>:0.0	This sets the environment to display all functions to this window.
8. On the second SUN5 session: **xrun &**	This initiates XRunner.

9. From the XRunner window, **click Record and select Start Recording - Analog.**	The record function of XRunner is now running.
10.**Click the File pulldown menu.**	If the File pulldown menu is clicked, two choices of Open and Exit will appear.
11.**Click Open**.	If Open is clicked, the file selection box will appear giving the user the option to open a log file. Note: At the file selection box, select only the log files.
12.**Double click /usr/testa/kmc..**	Backs up to directory /usr/testa.
13.**Double click /usr/testa/..**	Backs up to directory /usr.
14.**Double click /usr/..**	
15.Scroll down to /**tmp** and double click.	
16.In the files area in the fileDialogShell, there will be file ***/tmp/*.log***. Click on that file and **click OK**.	The log file opens in the ECS Browser Log window. C-MSS-18340
17.Select the first item (**Performance Poll**)and click OK.	The Event Field window appears. It will display three separate boxes: a) Host Information b) Event Field c) Performance Metric. C-MSS-18340
18.**Click close** to end Event Field window.	Event Field Window closes.
19.**Click Select pulldown menu for processing.**	When Select pulldown menu is clicked, three cascade pulldown menus of single select filters (by Category, Priority, Subsystem,) is displayed with the Custom Select option. C-MSS-18340
20.**Click 'Category'**	When 'Category' is selected, five cascade pulldown submenus are displayed: Fault, Performance, Security Alert, Topology Change, and Transaction. When one of the options is selected, all events of the selected option is displayed in the events box. C-MSS-18340
21.**Click 'Priority'**	When 'Priority' is selected, four cascade pulldown submenus are displayed: Low, Medium Low, Medium, and High. When one of the options is selected, all events of the selected option is displayed in the events box. C-MSS-18340
22.**Click 'Subsystem'**	When 'Subsystem' is selected, eight cascade pulldown submenus are displayed: CSS, DMS, DPS, DSS, INS, IOS, MSS, and Pdps. When one of the options is selected, all events of the selected option is displayed in the events box. C-MSS-18340
23.**From Select** pulldown menu, **click 'Custom Select'** or select the 'Custom Select' button at the bottom of the ECS Log Browser window.	When the 'Custom Select' is clicked, the custom select window is displayed. (Currently Category, Priority, Subsystem, and Type are available for testing. but ApplicationID, CSCI, InstanceID, Mode, ProcessID, and ProgramID functionally will be provided in Phase 3. C-MSS-18340
24.**Click cancel** to close window.	
25. **Click Sort** pulldown menu or Select pulldown menu for processing.	If Sort pulldown menu is clicked, six choices of single sort and a Custom Sort for multi-level sorting will appear: Time Stamp, Category, Type, Priority, Subsystem, CSCI. C-MSS-18340

26.**Click 'Time Stamp'**	All events will be listed in chronological order. Note: This process will take a few minutes to generate due to the size of the log. C-MSS-18340
27.**Click 'Category'**	All events will be grouped in categories. Note: This process will take a few minutes to generate due to the size of the log. C-MSS-18340
28.**Click 'Type'**	All events with same types will be grouped and listed. Note: This process will take a few minutes to generate due to the size of the log. C-MSS-18340
29.**Click 'Priority'**	All events with the same priorities will be grouped and listed. Note: This process will take a few minutes to generate due to the size of the log. C-MSS-18340
30.**Click 'Subsystem'**	All events will be grouped by susbsystems and listed. Note: This process will take a few minutes to generate due to the size of the log. C-MSS-18340
31.**Click 'CSCI'**	All events will be grouped by CSCI and listed. Note: This process will take a few minutes to generate due to the size of the log. C-MSS-18340
32.**From the Sort** pulldown menu, **click 'Custom Sort'** or click the 'Custom Sort' button at the bottom of the ECS Log Browser window.	When the Custom Sort is selected, the Custom Sort window appears and it gives the user the option to select multi-level of sorting in forward and reverse order of the following: Time Stamp, Category, Type, Priority, Subsystem, and CSCI. C-MSS-18340
33. **If Edit pulldown** menu is clicked, two choices of undo and reset will appear.	Undo = go back once reset = go back to the original (1st list)
34.**Select File - Exit**	The ECS Log Browser window closes.
35.On the **XRunner** window, **select Record - Stop Recording.**	This stops the XRunner animation.
36.**Select File - Save As**	This saves the animation to an XRunner script.
37.On the **XRunner** window, **select Open** and find the file that was just saved.	
38.**Select Replay - Animate**	XRunner script will then start processing.
39.**Select File - Exit to end XRunner.**	
40.End Test	

4.8.3.2 Test Case 2: ECS Log Browser for Application (TC022.002)

This test caseverifies the capability of the ECS Log Browser (ELB) to allow a user to browse the Application log file. The Application log file contains information that an ECS application records for its own purposes.

The capabilities tested include:

- providing a host, time period, and selection filter information.
- allowing an operator to selectively read a record from an Application log file.

- retrieving the requested data and display rows of brief description events.
- once displayed, options are given for sorting, additional filtering, editing, and displaying detail event fields.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MDA User Interface, MDA Services

Data: Application log, XRunner script

Tools: MsMdPhase2 (test driver), XRunner

Test Input:

Inputs to this test case include keyins to open and read a record from an Application log file.

Test Output:

Outputs to this test case include the description of an event field and an XRunner script.

Success Criteria:

This test is successful when the Application Log containing information on an ECS application can be browsed based on the selection filter information and requested data is retrieved, displayed, and sorted. The successful creation and execution of an XRunner script is also required.

Test Procedures:

Test Case ID: TC022.002	
Test Name: ECS Log Browser for Applications	
Test Steps:	Comments:
1. On the test machine (SUN5), set the environment variable of FileDir with *log at the end: **setenv FileDir <*usr/testa/kmc/*log*>**	This test cannot run on the HP machine in the mini-DAAC (msse4hp) with XRunner. At the present time, there isn't an HP license for XRunner.
2. Set the Display environment variable: **setenv DISPLAY <host ip address>:0.0**	
3.Change directory to where the test drivers reside: **cd** *</usr/testa/kmc>*	
4. Execute the test program: **MsMdPhase2 &**	The MsMdPhase2 application will run in the background. The msse3sun terminal will be in the interactive mode. The ECS Log Browser is displayed on the screen.
5. From the command line: **ps**	This will verify that the MsMdPhase2 process is running.

6. Open another SUN5 session: **select xterm from the pulldown menu**	This session will be used for XRunner.
7. On the second SUN5 session: **setenv DISPLAY <host ip address>:0.0**	This sets the environment to display all functions to this window.
8. On the second SUN5 session: **xrun &**	This initiates XRunner.
9. From the XRunner window, **click Record and select Start Recording - Analog.**	The record function of XRunner is now running.
10.**Click the File pulldown menu.**	If the File pulldown menu is clicked, two choices of Open and Exit will appear.
11.**Click Open.**	If Open is clicked, the file selection box will appear giving the user the option to open a log file. Note: At the file selection box, select only the log files.
12.**Double click *./usr/testa/kmc..***	Backs up to directory /usr/testa.
13.**Double click *./usr/testa/..***	Backs up to directory /usr.
14.**Double click *./usr/..***	
15.Scroll down to ***/tmp*** and **double click.**	
16.In the files area in the fileDialogShell, there will be file ***/tmp/*log.*** **Click on that file and click OK.**	The log file opens in the ECS Browser Log window.
17.**Select the first item** (**Application Poll**)and **click OK.**	The Event Field window appears. It will display three separate boxes: a) Host Information b) Event Field c) Performance Metric. C-MSS-18340
18.**Click close** to end Event Field window.	Event Field Window closes.
19.**Click Select pulldown menu for processing.**	When Select pulldown menu is clicked, three cascade pulldown menus of single select filters (by Category, Priority, Subsystem,) is displayed with the Custom Select option. C-MSS-18340
20.**Click 'Category'**	When 'Category' is selected, five cascade pulldown submenus are displayed: Fault, Performance, Secrutiy Alert, Topology Change, and Transaction. When one of the options are selected, all events of the selected option is displayed in the events box. C-MSS-18340
21.**Click 'Priority'**	When 'Priority' is selected, four cascade pulldown submenus are displayed: Low, Medium Low, Medium, and High. When one of the options are selected, all events of the selected option is displayed in the events box. C-MSS-18340
22.**Click 'Subsystem'**	When 'Subsystem' is selected, eight cascade pulldown submenus are displayed: CSS, DMS, DPS, DSS, INS, IOS, MSS, and Pdps. When one of the options are selected, all events of the selected option is displayed in the events box. C-MSS-18340
23.**From Select pulldown menu, click 'Custom Select' or select the 'Custom Select' button** at the bottom of the ECS Log Browser window.	When the 'Custom Select' is clicked, the custom select window is displayed. (Currently Category, Priority, Subsystem, and Type are available for testing. ApplicationID, CSCI, InstanceID, Mode, ProcessID, functionally will be provided in Phase 3.) C-MSS-18340

24.**Click cancel** to close window.	
25. **Click Sort pulldown menu or Select pulldown menu for processing**.	If Sort pulldown menu is clicked, six choices of single sort and a Custom Sort for multi-level sorting will appear: Time Stamp, Category, Type, Priority, Subsystem, CSCI. C-MSS-18340
26.**Click 'Time Stamp'**	All events will be listed in order of most recent time order. Note: This process will take a few minutes to generate due to the size of the log. C-MSS-18340
27.**Click 'Category'**	All events will be grouped in categories. Note: This process will take a few minutes to generate due to the size of the log. C-MSS-18340
28.**Click 'Type'**	All events with same types will be grouped and listed. Note: This process will take a few minutes to generate due to the size of the log. C-MSS-18340
29.**Click 'Priority'**	All events with the same priorities will be grouped and listed. Note: This process will take a few minutes to generate due to the size of the log. C-MSS-18340
30.**Click 'Subsystem'**	All events will be grouped by subsystems and listed. Note: This process will take a few minutes to generate due to the size of the log. C-MSS-18340
31.**Click 'CSCI'**	All events will be grouped by CSCI and listed. Note: This process will take a few minutes to generate due to the size of the log. C-MSS-18340
32.**From the Sort pulldown menu, click 'Custom Sort' or click the 'Custom Sort' button** at the bottom of the ECS Log Browser window.	When the Custom Sort is selected, the Custom Sort window appears and it gives the user the option to select multi-level of sorting in forward and reverse order of the following: Time Stamp, Category, Type, Priority, Subsystem, and CSCI. C-MSS-18340
33. If Edit pulldown menu is clicked, two choices of undo and reset will appear.	Undo = go back once reset = go back to the original (1st list)
34.**Select File - Exit**	The ECS Log Browser window closes.
35.On the **XRunner** window, **select Record - Stop Recording.**	This stops the XRunner animation.
36.**Select File - Save As**	This save the animation to an XRunner script.
37.On the **XRunner** window, **select Open and find the file that was just saved**.	
38.**Select Replay - Animate**	XRunner script will then start processing.
39.**Select File - Exit to end XRunner**.	
40.End Test	

4.8.4 Accountability Thread (TC023)

The purpose of the Accountability thread is to verify that the appropriate information contained within the MSS Accountability Management database is available and supported by the MSS Accountability Management Service. The service provides audit trail capability as a means to

verify the integrity of the system with the generation of a data audit trail from data collected from a variety of sources.

The functionality of the MSS Accountability Management Service includes the ability to:

- query user activity data, and data processing information stored in the Management Database.
- report audit information to M&O staff via the MUI service.
- retrieve user activity data (user id, type of user activity data items and date/time of the activity).
- retrieve data processing information logged for specified data items and from generated records.
- generate reports from collected management data.
- redirect reports to console, disk file and/or printer.

4.8.4.1 Test Case 1: Retrieving User Activity Data (TC023.001)

This test demonstrates the ability to retrieve user activity data information via the MSS Accountability Management service. The M&O staff will access the service given the contents of user id, type of user activity data items used (browsed, searched or ordered) and date/time of the activity from records generated from the Science Data Server, Data Processing and Client Subsystem.

Test Configuration:

Hardware: Local System Management (ECS) MSS Server (HP) and Workstation

Software: Accountability MUI, MDA, ECS Log Browser

Data: N/A

Tools: Sybase

Test Input:

User account and activity information (user id, data items)

Test Output:

Verification of inputs to accountability database regarding user activity data.

Success Criteria:

This test is considered successful when the MSS Accountability Management database retrieves the requested information regarding user activity data.

Test Procedures:

Test Case ID: TC023.001	
Test Name: Retrieving User Activity Data	
Test Steps:	Comments:
1. Login as a user capable of retrieving information for the Accountability Management Service.	
2. Set default to appropriate directory area.	
3. Display the Accountability menu page.	Accountability menu window appear.
4. Enter user id information	Access the Accountability database.
5. Select to retrieve user activity data information.	C-MSS-76000
6. Verify the retrieve screen is returned.	
7. Enter the activity data information to retrieve.	The activity selections are: (user id, type of user activity, data items used, date/time activity)
8. Select to access the database for desired activity data (browsed, searched or ordered).	C-MSS-76000
9. Verify the activity data returned contains the appropriate information.	
10. Exit out retrieve screen.	

4.8.4.2 Test Case 2: Retrieve/Query A Specified User or Data Item (TC023.002)

This test demonstrates the functionality to retrieve and query activities associated with a particular user or data item via the Management Data Access service.

Test Configuration:

Hardware: Local System Management (ECS) MSS Server (HP) and Workstation

Software: Accountability MUI, MDA, ECS Log Browser

Data: N/A

Tools: Sybase

Test Input:

User account and data order information (user id, data items)

Test Output:

Verifies inputs to Management Data Access regarding specified user or data item.

Success Criteria:

This test is successful when the management data access service queries and retrieves information from the database regarding a particular user or specified item.

Test Procedures:

Test Case ID: TC023.002	
Test Name: Retrieve/Query Specified User/Data	
Test Steps:	Comments:
1. Login as a user capable of retrieving information from the Accountability Management Service.	
2. Set default to appropriate directory area.	
3. Display the Accountability menu page.	Accountability menu window appear.
4. Enter user id information	Access the Accountability database.
5. Select to retrieve specified user.	
6. Enter specified user	
7. Select to query the MDA for specified user.	C-MSS-76010
8. Select to retrieve from the MDA service that specified user information.	C-MSS-76020 Given all activities associated with specified user or data item.
9. Verify the specified user information is returned.	
10.Select to retrieve particular data item.	
11.Enter particular data item.	
12.Select to query the MDA for particular data item.	C-MSS-76010
13.Select to retrieve from the MDA service that particular data item information.	C-MSS-76020 Given all activities associated with specified user or data item.
14.Verify the particular data information is returned.	
15.Exit out of screen.	

4.8.4.3 Test Case 3: Retrieve/Query Data Processing Information (TC023.003)

This test demonstrates the capability to retrieve data processing information, such as the instrument used, date/time of ingest or algorithm used (name and version), and date/time of data processing from records generated. The Management Data Access service queries for activities associated with a particular user or specified data item and retrieve all data processing information logged for specified data item.

Test Configuration:

Hardware: Local System Management (ECS) MSS Server (HP) and Workstation

Software: Accountability MUI, MDA, ECS Log Browser

Data: N/A

Tools: Sybase

Test Input:

User account and data information (user id, data items)

Test Output:

Verifies inputs to Management Data Access regarding specified user or data item.

Success Criteria:

This test is successful when the management data access service retrieves and queries the database for data processing information regarding a particular user or specified data item.

Test Procedures:

Test Case ID: TC023.003	
Test Name: Retrieve/Query Data Process Info	
Test Steps:	Comments:
1. Login as a user capable of retrieving information from the Accountability Management Service.	
2. Set default to appropriate directory area.	
3. Display the Accountability menu page.	Accountability menu window appear.
4. Enter user id information	Access the Accountability database.
5. Select to access Data Processing Information.	
6. Retrieve the data processing information.	C-MSS-77000 The data processing selections are: instrument used, date/time of ingest or algorithm used(name and version), date/time of processing.
7. Verify that the information is retrieved.	
8. Retrieve all data processing information logged for a specific data item.	C-MSS-77030 Retrieves data processing information logged for a specified data item.
9. Verify the specified data information is retrieved.	
10.Verify that the user can query the management database for all data processing information stored in the management database.	C-MSS-77010
11. Exit out of screen.	

4.8.4.4 Test Case 4: Data Audit Trail (TC023.004)

The purpose of the Data Audit test is to verify and report progress of orders as they are processed via the Accountability MUI service. This test ensures that upon completion of an order, a record of the completion is entered into the database, which includes the user who ordered the product, the product ordered, the media type requested, the name of the completed activity, and the duration of the time from placement of order to completion of order.

Test Configuration:

Hardware: Local System Management (ECS) MSS Server (HP) and Workstation

Software: Accountability MUI, MDA, ECS Log Browser

Data: N/A

Tools: Sybase

Test Input:

Placement of a data order and searching of logs.

Test Output:

Outputs to this test case include a printout (screen dump) of data processing information, queries based on test inputs and outputs of log data.

Success Criteria:

This test is successful when all of the desired data is reviewed for completeness, and the records are ingested to the database.

Test Procedures:

Test Case ID: TC023.004	
Test Name: Data Audit Trail	
Test Steps:	Comments:
1. Login as a user capable of retrieving information from the Accountability Management Service.	
2. Set default to appropriate directory area.	
3. Display the Accountability menu page.	Accountability menu window appear.
4. Enter user id information	Access the Accountability database.
5. Select to access data processing information.	
6. Retrieve the data processing information.	C-MSS-77000 The data processing selections are: instrument used, date/time of ingest or algorithm used(name and version), date/time of processing.
7. Verify that the user can query the management database for all data processing information stored in the management database.	C-MSS-77010
8. Search the logs to find data audit information.	Stored local history logs.
9. Report audit information to user.	C-MSS-76040 Report information via MUI service, ccmail, etc.
10.Exit out of audit screen.	

4.8.4.5 Test Case 5: Generate and Redirect Reports (TC023.005)

The purpose of this test is to demonstrate the capability to report collected management data on user activity data and data processing information. To redirect the generated reports to a console, disk file and printer.

Test Configuration:

Hardware: Local System Management (ECS) MSS Server (HP) and Workstation

Software: Accountability MUI, MDA, ECS Log Browser

Data: N/A

Tools: Sybase

Test Input:

Placement of collected data management information.

Test Output:

Outputs to this test case generates a report that can be directed to a console, disk file and printer based on test inputs and outputs of log data.

Success Criteria:

This test is successful when all of the desired data information is generated to a report file for redirection to a console, disk file and/or printer.

Test Procedures:

Test Case ID: TC023.005	
Test Name: Generate/Redirect a Report	
Test Steps:	Comments:
1. Login as a user capable of retrieving information from the Accountability Management Service.	
2. Set default to appropriate directory area.	
3. Display the Accountability menu page.	Accountability menu window appear.
4. Enter user id information	Access the Accountability database.
5. Select to retrieve user activity data information.	C-MSS-76000
7. Enter the activity data information to retrieve.	The activity selections are: user id, type of user activity, data items used, date/time activity.
8. Verify activity data information is retrieved.	
9. Select to generate a report.	C-MSS-77080 Collected from management database.
10. Verify report is generated.	
11. Print the generated report.	
12. Select to direct report : a) console b) disk file c) printer	C-MSS-77090
13.Select to retrieve data processing information.	
14.Retrieve the data processing information.	C-MSS-77000 The data processing selections are: instrument used, date/time of ingest or algorithm used(name and version), date/time of processing.
15.Verify data processing information is retrieved.	
16. Select to generate a report.	C-MSS-77080 Collected from management database.
17.Verify report is generated.	
18.Print the generated report.	

19.Select to direct report : a) console b) disk file c) printer	C-MSS-77090
20. Exit to generate a report.	

4.8.5 Change Request Manager Thread (TC024)

The Change Request Manager Service provides the functionality to compose, submit, coordinate, and track status of proposals to change ECS resources. The Change Request Manager Service is provided by the COTS tool Distributed Defect Tracking System (DDTS). DDTS, the Change Request Manager, maintains proposals for changing the configuration of the ECS and tracks their implementation. A sample of the types of change requests that exist are configuration change requests (CCR), non-conformance reports (NCR), deviations and waivers. Originators are notified when events occur relative to their requests. Reports containing lists of change requests, their descriptions, and their status are produced. The DDTS provides the capability of tracking and managing changes by providing change request records and accommodating resource change requests.

4.8.5.1 Test Case 1: Installation of Change Request Manager Test (TC024.001)

This test case verifies the custom installation procedures and scripts of the Change Request Manager (CRM) and of DDTS.

Test Configuration:

Hardware: LSM and SMC Workstation (msse3sun, msse7sun)

Software: Change Request Manager custom tar file

Data: Installation procedures

Tools: DDTS

Test Input:

Inputs to this test case include the DDTS installation guide and the installation procedures that properly set up the Change Request Manager, its classes, projects, and LSM to SMC site connection information.

Test Output:

The expected outputs of this test include installation and activation of DDTS, custom class and project installation, and site connection (communication) between the LSM CRM and the SMC CRM.

Success Criteria:

This test is successful when the user/installer receives e-mail notification of LSM and SMC CRM activation and communications. These e-mail notifications will include class and project subscriptions from LSM to SMC, and successful site communications. Execution of DDTS, and the submittal of a Configuration Change Request and a Non-Conformance Report at each site also deems this test successful.

Test Procedures:

Test Case ID: TC024.001	
Test Name: Installation of CRM Test	
Test Steps:	Comments
1. Log into LSM or SMC workstation and change directory to desired DDTS home.	
2. Extract Change Request Manager tar file. "**tar xvf <tar filename>**"	All elements, files and directories will be extracted from the tar file and stored in the DDTS home directory.
3. As **ddts** (account), run **ddtsinstall** and follow installation procedures according to DDTS Administrator's Manual.	**Ddtsinstall** will start all necessary DDTS daemons needed.
4. During **ddtsinstall**, when prompted for site code, enter "**SMCdd**" if on SMC workstation, or "**GSFcd**" if on LSM workstation.	For proper testing of custom installation procedures, site codes were mimicked in the mini-DAAC environment.
	When **ddtsinstall** is complete, the ECS standard projects must be established.
5. Using custom installation procedures, begin DDTS class/project customization.	The custom installation procedures will create ECS standard projects for the Change_Request, software, and hardware classes.
6. If installing on SMC workstation, DDTS administrator is "**smcuser**". If on LSM workstation DDTS administrator is "**lsmuser**".	Used as entries for DDTS administrator e-mail prompts.
7. When complete with custom installation, execute "**adminbug**". Enter **lsit** on each of the LSM and SMC workstations.	On SMC workstation will receive: **GSFcd: ddts@msse3sun**. On LSM workstation will receive: **SMCdd: ddts@msse7sun**
8. On LSM workstation, at "**adminbug**" prompt, enter "**lsub**".	Will display project subscriptions. **ECS_CHNG_REQ, ECS_NCR_SW, ECS_NCR_HW**
9. Enter "**quit**" at "**adminbug**" prompt.	Custom install procedures are complete.
10. Check DDTS administrator e-mail on both workstations (**lsmuser** - **msse3sun**, **smcuser** - **msse7sun**)	Confirmation of site connection and subscription will be mailed to each party.

4.8.5.2 Test Case 2: Compose/Submit a Configuration Change Request (CCR) (TC024.002)

This test case demonstrates the ability of a user to compose and submit a local and/or an ECS configuration change request (CCR) for the purpose of requesting system problems or enhancements.

Test Configuration:

Hardware: LSM and SMC workstation (msse3sun, msse7sun)

Software: Change Request Manager custom configuration

Data: TBD

Tools: DDTS

Test Input:

Inputs to this test case include DDTS commands and information required to submit a CCR.

Test Output:

The expected outputs of this test include a newly generated change request which has been distributed as assigned (e-mailed to assigned users) and maintains a current approval status.

Success Criteria:

This test is successful when a user is able to electronically draft a change request, submit it, and track it through the approval process. Upon submittal, the user will receive an e-mail message containing the completed CCR form and its status within DDTS.

Test Procedures:

Test Case ID: TC024.002	
Test Name: Compose/Submit a CCR	
Test Steps:	Comments:
1. Log into workstation and execute DDTS. "**/data/CRM/ddts/bin/xddts**"	DDTS main window will appear in top right hand corner of window.
2. Click on "**Select**" from the DDTS main window menu bar. Click on "**Select Project, Class & State**".	The **Select Project, Class & State** dialog window will pop up. In this window the user will select the class, project, and states they wish to view.
3. Click on "**Change Class**" button, and select "**Change_Request**".	**Change_Request** will appear in Class Name field. These Change Requests are associated with the associated site Change Request Manager.
4. Click on desired project, either "**ECS_CHNG_REQ**" or "**GSFC_CHNG_REQ**".	The chosen project will become highlighted. If the "**Select All**" button were pressed, both the **ECS_CHNG_REQ** and/or the **GSFC_CHNG_REQ** would be selected.
5. Click on "**Select All**" button under the **States** column.	This will select all the states. This will allow the user to view the CCRs as they transition states in DDTS.
6. Click on "**OK**" and return to main window.	Saves selected class, project, and states for current session.
7. Click on "**Submit**" button to begin.	The "**Submit a New Change Request**" form will be displayed in the Record window.
8. Enter "**Change_Request**" at class prompt.	
9. Enter "**GSFC_CHNG_REQ**" or "**ECS_CHNG_REQ**" at project name prompt.	Loads **GSFC_CHNG_REQ** or **ECS_CHNG_REQ** form into Record window.

10. At each prompt enter information based on category. Enter a valid entry for each "one of" or multiple entry fields. Also enter an invalid entry in each of the "one of" or multiple entry fields.	Since the data fields are of data type character, any character is a valid entry. The "one of" fields and multiple entry fields will not allow invalid entries.
11. Click on "**Change_State**" from Record windows menu bar, and select "**Assign-Eval**".	This will transition the state of the CCR to evaluation, and will change to page 2 of the CCR.
12. At each prompt enter information based on category. Enter a valid entry for each of "one of" or multiple entry fields. Also enter an invalid entry in each of the "one of" and multiple entry fields.	Since the data fields are of data type character, any character is a valid entry. The "one of" and multiple entry fields will not allow entries other than requested listing.
13. Click on "**Change_State**" from Record windows menu bar, and select "**Assign-Implement**".	This will transition the state of the CCR to implementation, and will change to page 3 of the CCR.
14. At each prompt enter information based on category. Enter a valid entry for each of "one of" or multiple entry fields. Also enter an invalid entry in each of the "one of" and multiple entry fields.	Since the data fields are of data type character, any character is a valid entry. The "one of" and multiple entry fields will not allow entries other than requested listing.
15. When complete, the form will return to page 1. Click on "**Commit**" button in DDTS main window.	C-MSS-40600 SMC-2515#A This commits the CCR form into the database.
16. Check e-mail for incoming DDTS messages.	User will receive an e-mail from DDTS informing them about the CCR that was submitted, and its current state (**Assign-Implement**).

4.8.5.3 Test Case 3: Compose/Submit a Hardware Non-Conformance Report (TC024.003)

This test case demonstrates the ability of a user to compose and submit a local and/or ECS non-conformance request against hardware defects.

Test Configuration:

Hardware: LSM and SMC workstation (msse3sun, msse7sun)

Software: Change Request Manager custom configuration

Data: TBD

Tools: DDTS

Test Input:

Inputs to this test case include DDTS commands and information required to submit a hardware non-conformance report.

Test Output:

The expected outputs of this test include a newly generated NCR which has been distributed as assigned (e-mailed to assigned users) and maintains a current approval status.

Success Criteria:

This test is successful when a user is able to electronically draft a NCR, submit it, and track it through the approval process. Upon submittal, the user will receive an e-mail message containing the completed NCR form and its status within DDTS.

Test Procedures:

Test Case ID: TC024.003	
Test Name: Compose/Submit a Hardware NCR	
Test Steps:	Comments:
1. Log into workstation and execute DDTS. "**/data/CRM/ddts/bin/xddts**"	DDTS main window will appear in top right hand corner of window.
2. Click on "**Select**" from the DDTS main window menu bar. Click on "**Select Project, Class & State**".	The **Select Project, Class & State** dialog window will pop up. In this window the user will select the class, project, and states they wish to view.
3. Click on "**Change Class**" button, and select "**hardware**".	"**hardware**" will appear in Class Name field. This hardware class is associated with the associated site Change Request Manager.
4. Click on desired project, either "**ECS_NCR_HW**" or "**GSFC_NCR_HW**".	The chosen project will become highlighted. If the "**Select All**" button were pressed, both the **ECS_NCR_HW** and/or the **GSFC_NCR_HW** would be selected.
5. Click on "**Select All**" button under the States column.	This will select all the states. This will allow the user to view the NCRs as they transition states in DDTS.
6. Click on "**OK**" and return to main window.	Save selected class, project, and states for current session.
7. Click on "**Submit**" button to begin.	The "**Submit a New Defect**" form will be displayed in the Record window.
8. Enter "**hardware**" at class prompt.	
9. Enter "**GSFC_NCR_HW**" or "**ECS_NCR_HW**" at project name prompt.	Loads **GSFC_NCR_HW** or **ECS_NCR_HW** form into Record window.
10. At each prompt enter information based on category. Enter a valid entry for each "one of" or multiple entry fields. Also enter an invalid entry in each of the "one of" or multiple entry fields.	Since the data fields are of data type character, any character is a valid entry. The "one of" fields and multiple entry fields will not allow invalid entries.
11. When complete, click on "**Commit**" button in DDTS main window.	C-MSS-40600 SMC-2515#A The commits the NCR into the database.
12. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**New**).

4.8.5.4 Test Case 4: Compose/Submit a Software Non-Conformance Report (TC024.004)

This test case demonstrates the ability of a user to compose and submit a local and/or ECS non-conformance request against software defects or enhancements.

Test Configuration:

Hardware: LSM and SMC workstation (msse3sun, msse7sun)

Software: Change Request Manager custom configuration

Data: TBD

Tools: DDTS

Test Input:

Inputs to this test case include DDTS commands and information required to submit a software non-conformance report.

Test Output:

The expected outputs of this test include a newly generated NCR which has been distributed as assigned (e-mailed to assigned users) and maintains a current approval status.

Success Criteria:

This test is successful when a user is able to electronically draft a NCR, submit it, and track it through the approval process. Upon submittal, the user will receive an e-mail message containing the completed NCR form and its status within DDTS.

Test Procedures:

Test Case ID: TC024.004	
Test Name: Compose/Submit a Software NCR	Comments:
Test Steps:	
1. Log into workstation and execute DDTS. "**/data/CRM/ddts/bin/xddts**"	DDTS main window will appear in top right hand corner of window.
2. Click on "**Select**" from the DDTS main window menu bar. Click on "**Select Project, Class & State**".	The **Select Project, Class & State** dialog window will pop up. In this window the user will select the class, project, and states they wish to view.
3. Click on "**Change Class**" button, and select "**software**".	"**software**" will appear in Class Name field. This software class is associated with the associated site Change Request Manager.
4. Click on desired project, either "**ECS_NCR_SW**" or "**GSFC_NCR_SW**".	The chosen project will become highlighted. If the "**Select All**" button were pressed, both the **ECS_NCR_SW** and/or the **GSFC_NCR_SW** would be selected.
5. Click on "**Select All**" button under the **States** column.	This will select all the states. This will allow the user to view the NCRs as they transition states in DDTS.
6. Click on "**OK**" and return to main window.	Save selected class, project, and states for current session.
7. Click on "**Submit**" button to begin.	The "**Submit a New Defect**" form will be displayed in the Record window.
8. Enter "**software**" at the class prompt.	
9. Enter "**GSFC_NCR_SW**" or "**ECS_NCR_SW**" at project name prompt.	Loads **GSFC_NCR_SW** or **ECS_NCR_SW** form into Record window.

10. At each prompt enter information based on category. Enter a valid entry for each "one of" or multiple entry fields. Also enter an invalid entry in each of the "one of" or multiple entry fields.	Since the data fields are of data type character, any character is a valid entry. The "one of" fields and multiple entry fields will not allow invalid entries.
11. When complete, click on "**Commit**" button in DDTS main window.	C-MSS-40600 SMC-2515#A The commits the NCR into the database.
12. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**New**).

4.8.5.5 Test Case 5: State Transition of a Configuration Change Request (TC024.005)

This test case demonstrates the ability to transition a configuration change request through the established states as defined in documentation, and assign the CCR to a responsible engineer.

Test Configuration:

Hardware: LSM/SMC Workstations (msse3sun, msse7sun)

Software: Change Request Manager custom configuration

Data: N/A

Tools: DDTS

Test Input:

Inputs to this test case include DDTS commands and information required transition a CCR through the desired states.

Test Output:

The expected outputs of this test include a state transition of a previously entered configuration change request, and an e-mail message reported to the assigned responsible engineer, and to the user performing the transition.

Success Criteria:

This test is successful when the user is able to transition a configuration change request, re-submit the request, view and verify the state transition from the history log, and receive e-mail notification of state transition.

Test Procedures:

Test Case ID: TC024.005	
Test Name: State Transition of a CCR	
Test Steps:	Comments
1. Log into workstation and execute DDTS. "**/data/CRM/ddts/bin/xddts**"	DDTS main window will appear in top right hand corner of window.

2. Click on "**Select**" from the DDTS main window menu bar. Click on "**Select Project, Class & State**".	The **Select Project, Class & State** dialog window will pop up. In this window the user will select the class, project, and states they wish to view.
3. Click on "**Change Class**" button, and select "**Change_Request**".	**Change_Request** will appear in Class Name field. These Change Requests are associated with the associated site Change Request Manager.
4. Click on desired project, either "**ECS_CHNG_REQ**" or "**GSFC_CHNG_REQ**".	The chosen project will become highlighted. If the "**Select All**" button were pressed, both the **ECS_CHNG_REQ** and/or the **GSFC_CHNG_REQ** would be selected.
5. Click on "**Select All**" button under the States column.	This will select all the states. This will allow the user to view the CCRs as they transition states in DDTS.
6. Click on "**OK**" and return to main window.	Save selected class, project, and states for current session.
7. Select a previously submitted CCR from the DDTS main window listing.	The selected CCR will be displayed in the Record window. If the CCR that was selected is currently in an **ASSIGNED-IMPLEMENT** state, skip to step 14.
8.. Click on "**Change_State**" from Record windows menu bar, and select "**Assign-Eval**".	This will transition the state of the CCR to evaluation, and will change to page 2 of the CCR.
9.. At each prompt enter information based on category. Enter a valid entry for each of "one of" or multiple entry fields. Also enter an invalid entry in each of the "one of" and multiple entry fields.	Since the data fields are of data type character, any character is a valid entry. The "one of" and multiple entry fields will not allow entries other than requested listing.
10. Click on the "**Commit**" button in the DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the CCR to the database, and updates the DDTS main window to reflect the state change.
11. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the CCR that was submitted, and its current state (**Assign-Eval**).
12. Click on "**Change_State**" from Record windows menu bar, and select "**Assign-Implement**".	This will transition the state of the CCR to implementation, and will change to page 3 of the CCR.
13. At each prompt enter information based on category. Enter a valid entry for each of "one of" or multiple entry fields. Also enter an invalid entry in each of the "one of" and multiple entry fields.	Since the data fields are of data type character, any character is a valid entry. The "one of" and multiple entry fields will not allow entries other than requested listing.
14. Click on the "**Commit**" button in the DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the CCR to the database, and updates the DDTS main window to reflect the state change.
15. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the CCR that was submitted, and its current state (**Assign-Implement**).
15. Click on "**Change_State**" from the menu bar on the Record window. Select "**R Implement**".	Notice that the Status field on page 1 of the CCR has changed to '**Implemented**'.
16. Click on the "**Commit**" button in the DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the CCR to the database, and updates the DDTS main window to reflect the state change.

17. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the CCR that was submitted, and its current state (**Implement**).
18. Click on "**Change_State**" from the menu bar on the Record window. **Select 'T Assign-Verify**'.	The "**TESTING INFORMATION**" column will appear on page 3 of the CCR form.
19. Enter information related to the specified prompts.	The Organization attribute is a 'one of' field. The list contains ASF, EDC, EOC, GSFC, JPL, LaRC, NSIDC, ORNL, SMC for a total of 9 tests. Notice that the status field on page 1 of the CCR form has changed to '**Assign-Verified**'.
20. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the CCR to the database, and updates the DDTS main window to reflect the state change.
21. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the CCR that was submitted, and its current state (**Assign-Verify**).
22. Click on "**Change_State**" from the menu bar on the Record window. Select '**V Verify**'	The "**VERIFICATION INFORMATION**" column will appear on page 3 of the CCR form.
23. Enter '**Test Status**' information.	The '**Test Status**' field has 2 choices, Pass, Fail, for a total of 2 tests.
24. If the user selects to add an Enclosure, click on "**Modify**" from menu bar on the Record window, and select "**Add Enclosure**".	An Enclosure window will appear. Enter information.
25. When complete, click on "**File**" from the menu bar on the Editor window, and select "**Save Changes & Dismiss Editor**".	Saves Enclosure and prompts for title of enclosure.
26. Enter the name of the enclosure and click on "**OK**".	The enclosure will appear in the Enclosures window.
27. At '**Enclosure Added**' prompt enter Y or N.	Notice that the status field on page 1 of the CCR form has changed to 'Verified'
28. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	C-MSS-40690, C-MSS-40730, C-MSS-40750 SMC-2520#A Stores the transition of the CCR to the database, and updates the DDTS main window to reflect the state change.
29. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the CCR that was submitted, and its current state (**Verify**).
	From this point, the last four states, '**C Close**', '**D Duplicate**', '**P Defer**', and '**F Forward**', have to be tested separately. The '**Forward**' state is a separate test case.
30. Double click on the "**History**" icon in the Enclosures window.	View the contents of the enclosure and verify state transition and user information for validity.
To **CLOSE** a CCR:	
31. To close a CCR, click on "**Change_State**" from the menu bar in the Record window. Select '**C Close**'.	The "**CLOSING INFORMATION**" rows will appear on page 3 of the CCR form.
32. Enter information related to the specified prompts.	The Organization attribute is a 'one of' field. The list contains ASF, EDC, EOC, GSFC, JPL, LaRC, NSIDC, ORNL, SMC for a total of 9 tests. Notice that the status field on page 1 of the CCR form has changed to 'Closed'.

33. Click on "**Commit**" button in DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the CCR to the database, and updates the DDTS main window to reflect the state change.
34. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the CCR that was submitted, and its current state (**Closed**).
35. Double click on the "**History**" icon in the Enclosures window.	C-MSS-40730 View the contents of the enclosure and verify state transition and user information for validity.
To **DEFER** a CCR:	
31. To defer a CCR, click on "**Change_State**" from the menu bar in the Record window. Select '**P Defer**'.	The '**Until**' attribute will appear on page 1 of the CCR form.
32. The '**Until**' field is a date field, enter a date.	Notice the status field on page 1 of the CCR form has changed to "**Deferred**".
33. Click on "**Commit**" button in DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the CCR to the database, and updates the DDTS main window to reflect the state change.
34. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the CCR that was submitted, and its current state (**Deferred**).
35. Double click on the "**History**" icon in the Enclosures window.	C-MSS-40730 View the contents of the enclosure and verify state transition and user information for validity.
To set a CCR as a **DUPLICATE**:	Note, cannot DUPLICATE a CLOSED CCR.
31. To transition a CCR to Duplicate, click on "**Change_State**" from the menu bar in the Record window. Select '**D Duplicate**'.	The '**Duplicate of CCR#**' attribute will appear on page 1 of the CCR form.
32. Enter a CCR# which this CCR is a Duplicate of.	If an invalid CCR# is entered, DDTS will inform that the CCR# does not exist and requests valid CCR# be entered. When complete, notice Status field on page 1 of the CCR form has changed to 'Duplicated'.
33. Click on "**Commit**" button in DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the CCR to the database, and updates the DDTS main window to reflect the state change.
34. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the CCR that was submitted, and its current state (**Duplicated**).
35. Double click on the "**History**" icon in the Enclosures window.	C-MSS-40730 View the contents of the enclosure and verify state transition and user information for validity.

4.8.5.6 Test Case 6: State Transition of a Non-Conformance Report (TC024.006)

This test case demonstrates the ability to transition an NCR through the established states as defined in documentation, and assign the NCR to a responsible engineer.

Test Configuration:

Hardware: LSM/SMC Workstation (msse3sun, msse7sun)

Software: Change Request Manager custom configuration

Data: N/A

Tools: DDTS

Test Input:

Inputs to this test case include DDTS commands and information required to transition a NCR through the desired states.

Test Output:

The expected outputs of this test include a state transition of a previously entered non-conformance report, and an e-mail message reported to the assigned responsible engineer and to the user performing the transition.

Success Criteria:

This test is successful when the user is able to transition a non-conformance report, re-submit the report, view and verify the state transitions from the history log, and receive e-mail notification of state transition.

Test Procedures:

Test Case ID: TC024.006	
Test Name: State Transition of a non-Conformance Report	
Test Steps:	Comments:
1. Log into workstation and execute DDTS. "**/data/CRM/ddts/bin/xddts**"	DDTS main window will appear in top right hand corner of window.
2. Click on "**Select**" from the DDTS main window menu bar. Click on "**Select Project, Class & State**".	The **Select Project, Class & State** dialog window will pop up. In this window the user will select the class, project, and states they wish to view.
3. Click on "**Change Class**" button, and select "**software**" or "**hardware**".	User selection will appear in the Class Name field. These NCRs are associated with the associated site Change Request Manager.
4. Click on desired project, either "**ECS_NCR_?W**" or "**GSFC_NCR_?W**".	The chosen project will become highlighted. If the "**Select All**" button were pressed, both projects would be selected. NOTE, the "**?**" denotes either an "**S**" or "**H**" depending on the class selected.
5. Click on "**Select All**" button under the **States** column.	This will select all the states. This will allow the user to view the NCRs as they transition states in DDTS.
6. Click on "**OK**" and return to main window.	Save selected class, project, and states for current session.
7. Select a previously submitted NCR from the DDTS main window listing.	The selected NCR will be displayed in the Record window.
If transitioning **HARDWARE** class:	
8. Click on "**Change_State**" from the menu bar on the Record window. Select **"A Assign-Eval**".	The "**ANALYSIS INFORMATION**" column will appear in the NCR form.

9. Enter information related to the specified prompts.	
10. Click on the "**Commit**" button in the DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **ASSIGNED-EVAL**.
11. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**ASSIGNED-EVAL**).
12. Click on "**Change_State**" from the menu bar on the Record window. Select "**O Assign-Fix**".	Additional information will appear below the "**ANALYSIS INFORMATION**" column.
13. Enter information related to the specified prompts.	The Failure Rating attribute is a 'one of' field. The list 1, 2, 3, 4, 5 for a total of 5 tests.
14. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **ASSIGNED_FIX**.
15. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**ASSIGNED-FIX**).
16, Click on "**Change_State**" from the menu bar on the Record window. Select "**R Fix**".	The NCR will transition to page 2, and the "**RESOLUTION INFORMATION**" column will appear.
17. Enter information related to the specified prompts.	
18. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **FIXED**.
19. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**FIXED**).
20. Click on "**Change_State**" from the menu bar on the Record window. Select "**T Assign-Verify**".	The NCR will transition to page 3, and the "**TESTING INFORMATION**" column will appear.
21. Enter information related to the specified prompts.	
22. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **ASSIGN-VERIFY**.
23. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**ASSIGNED-VERIFY**).
24. Click on "**Change_State**" from the menu bar on the Record window. Select "**V Verify**".	The NCR will transition to page 3, and the "**VERIFICATION INFORMATION**" column will appear.
25. Enter information related to the specified prompts.	Test Status attribute has 2 options, test each for completeness.

26. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **VERIFIED**.
27. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**VERIFIED**).
28. To **CLOSE** an NCR, Click on "**Change_State**" from the menu bar on the Record window. Select "**C Close**".	The NCR will transition to page 3, and the "**CLOSING INFORMATION**" column will appear.
29. Enter information related to the specified prompts.	
30. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **CLOSED**.
31. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**CLOSED**).
32. Double click on the "**History**" icon in the Enclosures window.	View the contents of the enclosure and verify state transition and user information for validity.
33. To **DUPLICATE** an NCR, Click on "Change_State" from the menu bar on the Record window. Select "**D Duplicate**".	If, "**ANALYSIS INFORMATION**" has not been entered, Enter information now. Note, cannot duplicate a CLOSED NCR.
34. Enter an NCR# which this NCR is a duplicate of.	If an invalid NCR# is entered, DDTS will inform that the NCR# does not exist and requests valid NCR# be entered.
35. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **DUPLICATE**.
36. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**DUPLICATED**).
37. Double click on the "**History**" icon in the Enclosures window.	C-MSS-40730 View the contents of the enclosure and verify state transition and user information for validity.
If transitioning **SOFTWARE** class:	
8. Click on "**Change_State**" from the menu bar on the Record window. Select "**A Assign-Eval**".	The "**ANALYSIS INFORMATION**" column will appear in the NCR form.
9. Enter information related to the specified prompts.	
10. Click on the "**Commit**" button in the DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **ASSIGNED-EVAL**.
11. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**ASSIGNED-EVAL**).
12. Click on "**Change_State**" from the menu bar on the Record window. Select "**O Assign-Fix**".	Additional information will appear below the "**ANALYSIS INFORMATION**" column.

13. Enter information related to the specified prompts.	Note, the "**Affects Subsystem**", "**CSCI**", "**Problem Type**", "**Recommended Change**", and "**Phase Problem Caused**" are all one-of fields. Test each option for completion.
14. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **ASSIGNED_FIX**.
15. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**ASSIGNED-FIX**).
16, Click on "**Change_State**" from the menu bar on the Record window. Select "**R Fix**".	The NCR will transition to page 2, and the "**RESOLUTION INFORMATION**" column will appear.
17. Enter information related to the specified prompts.	Note, "**Resolution**", and "**Phase Problem Fixed**" attributes are one-of fields. Test each option for completion.
18. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **FIXED**.
19. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**FIXED**).
20. Click on "**Change_State**" from the menu bar on the Record window. Select "**T Assign-Verify**".	The NCR will transition to page 2, and the "**TESTING INFORMATION**" column will appear.
21. Enter information related to the specified prompts.	
22. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **ASSIGN-VERIFY**.
23. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**ASSIGNED-VERIFY**).
24. Click on "**Change_State**" from the menu bar on the Record window. Select "**V Verify**".	The NCR will transition to page 2, and the "**VERIFICATION INFORMATION**" column will appear.
25. Enter '**Test Status**' information.	The '**Test Status**' field has 2 choices, Pass, Fail, for a total of 2 tests.
26. If the user selects to add an Enclosure, click on "**Modify**" from menu bar on the Record window, and select "**Add Enclosure**".	An Enclosure window will appear. Enter information.
27. When complete, click on "**File**" from the menu bar on the Editor window, and select "**Save Changes & Dismiss Editor**".	Saves Enclosure and prompts for title of enclosure.
28. Enter the name of the enclosure and click on "**OK**".	The enclosure will appear in the Enclosures window.
29. At '**Enclosure Added**' prompt enter Y or N.	Notice that the status field on page 1 of the CCR form has changed to '**Verified'**

30. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	C-MSS-40690, C-MSS-40730, C-MSS-40750 SMC-2520#A Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **VERIFIED**.
31. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**VERIFIED**).
32. To **CLOSE** an NCR, Click on "**Change_State**" from the menu bar on the Record window. Select "**C Close**".	The NCR will transition to page 2, and the "**CLOSING INFORMATION**" column will appear.
31. Enter information related to the specified prompts.	
32. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **CLOSED**.
33. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**CLOSED**).
34. Double click on the "**History**" icon in the Enclosures window.	C-MSS-40730 View the contents of the enclosure and verify state transition and user information for validity.
35. To **DUPLICATE** an NCR, Click on "**Change_State**" from the menu bar on the Record window. Select "**D Duplicate**".	If, "**ANALYSIS INFORMATION**" has not been entered, Enter information now. Note, cannot duplicate a CLOSED NCR.
36. Enter an NCR# which this NCR is a duplicate of.	If an invalid NCR# is entered, DDTS will inform that the NCR# does not exist and requests valid NCR# be entered.
37. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	C-MSS-40750 Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **DUPLICATE**.
38. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**DUPLICATED**).
39. Double click on the "**History**" icon in the Enclosures window.	C-MSS-40730 View the contents of the enclosure and verify state transition and user information for validity.

4.8.5.7 Test Case 7: Modification of Stored CCR Test (TC024.007)

This test case demonstrates the ability to recall a previously submitted Configuration Change Request and modify any of its contents.

Test Configuration:

Hardware: LSM/SMC Workstations (msse3sun, msse7sun)

Software: Change Request Manager custom configuration

Data: previously submitted CCR

Tools: DDTS

Test Input:

Inputs to this test case include DDTS commands and information required submit a CCR.

Test Output:

The expected outputs of this test include an entry(s) in the history enclosure of modifications and the appearance of a new enclosure attached to a previously submitted CCR. User will also receive an e-mail of the modifications and current state of the CCR.

Success Criteria:

This test is successful when the user is able to view and verify the new contents of the CCR, including the history enclosure and the new attached enclosure. The user will also receive an e-mail notice including the CCR and its modifications.

Test Procedures:

Test Case ID: TC024.007	
Test Name: Modification of Stored CCR Test	
Test Steps:	Comments:
1. Log into workstation and execute DDTS. "**/data/CRM/ddts/bin/xddts**"	DDTS main window will appear in top right hand corner of window.
2. Click on "**Select**" from the DDTS main window menu bar. Click **on "Select Project, Class & State**".	The **Select Project, Class & State** dialog window will pop up. In this window the user will select the class, project, and states they wish to view.
3. Click on "**Change Class**" button, and select "**Change_Request**".	**Change_Request** will appear in Class Name field. These Change Requests are associated with the associated site Change Request Manager.
4. Click on "**Select All**" button under the Projects column.	Both the **ECS_CHNG_REQ** and the **GSFC_CHNG_REQ** projects will be selected.
5. Click on "**Select All**" button under the States column.	This will select all the states. This will allow the user to view the CCRs as they transition states in DDTS.
6. Click on "**OK**" and return to main window.	Save selected class, project, and states for current session.
7. If on an LSM workstation, select a previously submitted SMC CCR from the DDTS main window.	The selected CCR will be displayed in the Record window. Notice "**Remote Record - Modification not allowed**" message on bottom of DDTS main window.
8. Select a previously submitted local CCR from the DDTS main window.	The selected CCR will be displayed in the Record window.
9. Click on "**Modify**" from the menu bar on the Record window. Select "**Modify Record**".	C-MSS-40610 The user is now able to modify the fields in the CCR form.

10. Change each field in the CCR form.	Note - some fields are "one of" selections. Since each of these selections were tested in TC024.002, user does not need to test all selections.
11. When completed changing CCR fields, click on "**Modify**" from the menu bar on the Record window. Select "**Add Enclosure**".	This allows the user to add information to the CCR without changing its state or original contents if so desired.
12. Enter text into enclosure, and when complete, click on "**File**" from the menu bar on the Enclosure window, and select "**Save Changes & Dismiss Editor**".	C-MSS-40670 SMC-2520#A
13. Enter the name of the enclosure and click on "**OK**"	Saves enclosure as new name and displays icon of enclosure in the Enclosures window.
14. To change or edit an already existing enclosure, double click on the enclosure icon.	The enclosure will be displayed for viewing.
15. Click on "**Edit Enclosure**".	Allows user to edit contents of enclosure.
16. When completed editing, click on "**File**" from the Enclosure window, and select "**Save Changes & Dismiss Editor**".	C-MSS-40670 SMC-2520#A Changes will be stored into enclosure and enclosure will be minimized to icon in Enclosures window.
17. Click on "**Commit**" button in DDTS main window.	C-MSS-40670, C-MSS-40690 SMC-2515#A Stores the modifications of the CCR to the database.
18. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the CCR that was submitted, and its current state.
19. Double click on "**History**" log in Enclosures window.	C-MSS-40730 Verify field modifications, new enclosure, and edited enclosure has been recorded.

4.8.5.8 Test Case 8: Modification of Stored NCR Test (TC024.008)

This test case demonstrates the ability to recall a previously submitted Non-Conformance Report and modify its contents.

Test Configuration:

Hardware: LSM/SMC Workstations (msse3sun, msse7sun)

Software: Change Request Manager custom configuration

Data: previously submitted NCR

Tools: DDTS

Test Input:

Inputs to this test case include DDTS commands and information required transition a NCR through the desired states.

Test Output:

The expected outputs of this test include an entry(s) in the history enclosure of modifications and the appearance of a new enclosure attached to a previously submitted NCR. User will also receive an e-mail of the modifications and current state of the NCR.

Success Criteria:

This test is successful when the user is able to view and verify the new contents of the NCR, including the history enclosure and the new attached enclosure. The user will receive an e-mail notification including the NCR and its modifications.

Test Procedures:

Test Case ID: TC024.008	
Test Name: Modification of Stored NCR Test	
Test Steps:	Comments:
1. Log into workstation and execute DDTS. "**/data/CRM/ddts/bin/xddts**"	DDTS main window will appear in top right hand corner of window.
2. Click on "**Select**" from the DDTS main window menu bar. Click on "**Select Project, Class & State**".	The **Select Project, Class & State** dialog window will pop up. In this window the user will select the class, project, and states they wish to view.
3. Click on "**Change Class**" button, and select "**software**" or "**hardware**" class.	User selection will appear in Class Name field. These software or hardware NCRs are associated with the associated site Change Request Manager.
4. Click on "**Select All**" button under the Projects column.	Both the **ECS_NCR_?W** and the **GSFC_NCR_?W** projects will be selected. (The **?** denotes either **S** for software or **H** for hardware.)
5. Click on "**Select All**" button under the States column.	This will select all the states. This will allow the user to view the NCRs as they transition states in DDTS.
6. Click on "**OK**" and return to main window.	Save selected class, project, and states for current session.
7. If on an LSM workstation, select a previously submitted SMC NCR from the DDTS main window.	The selected NCR will be displayed in the Record window. Notice "**Remote Record - Modification not allowed**" message on bottom of DDTS main window.
8. Select a previously submitted local NCR from the DDTS main window.	The selected NCR will be displayed in the Record window.
9. Click on "**Modify**" from the menu bar on the Record window. Select "**Modify Record**".	C-MSS-40610 The user is now able to modify the fields in the NCR form.
10. Change each field in the NCR form.	Note - some fields are "one of" selections. Since each of these selections were tested in TC024.003 and TC024.004, user does not need to test all selections.
11. When completed changing NCR fields, click on "**Modify**" from the menu bar on the Record window. Select "**Add Enclosure**".	This allows the user to add information to the NCR without changing its state or original contents if so desired.

12. Enter text into enclosure, and when complete, click on "**File**" from the menu bar on the Enclosure window, and select "**Save Changes & Dismiss Editor**".	C-MSS-40670 SMC-2520#A
13. Enter the name of the enclosure and click on "**OK**"	Saves enclosure as new name and displays icon of enclosure in the Enclosures window.
14. To change or edit an already existing enclosure, double click on the enclosure icon.	The enclosure will be displayed for viewing.
15. Click on "**Edit Enclosure**".	Allows user to edit contents of enclosure.
16. When completed editing, click on "**File**" from the Enclosure window, and select "**Save Changes & Dismiss Editor**".	C-MSS-40670 SMC-2520#A Changes will be stored into enclosure and enclosure will be minimized to icon in Enclosures window.
17. Click on "**Commit**" button in DDTS main window.	C-MSS-40670, C-MSS-40690 SMC-2515#A Stores the modifications of the NCR to the database.
18. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state.
19. Double click on "**History**" log in Enclosures window.	C-MSS-40730 Verify field modifications, new enclosure, and edited enclosure has been recorded.

4.8.5.9 Test Case 9: Forward Local NCR/CCR to SMC (TC024.009)

This test case demonstrates the ability to forward a locally submitted NCR or CCR to the SMC Change Request Manager.

Test Configuration:

Hardware: LSM and SMC workstation (msse3sun, msse7sun)

Software: Change Request Manager custom configuration

Data: N/A

Tools: DDTS

Test Input:

Inputs to this test case include a previously submitted non-conformance report and configuration change request and DDTS commands.

Test Output:

The expected outputs of this test include the NCR/CCR forwarded to the SMC Change Request Manager and e-mail notification.

Success Criteria:

This test is successful when the forwarded NCR/CCR is visible to the Change Request Manager at the local site and SMC, the history log verifies state transition, and an e-mail notification to the responsible engineer at the SMC.

Test Procedures:

Test Case ID: TC024.009	
Test Name: Forward Local NCR/CCR to SMC	
Test Steps:	Comments:
1. Log into LSM workstation and execute DDTS. "**/data/CRM/ddts/bin/xddts**"	DDTS main window will appear in top right hand corner of window.
2. Click on "**Select**" from the DDTS main window menu bar. Click on "**Select Project, Class & State**".	The **Select Project, Class & State** dialog window will pop up. In this window the user will select the class, project, and states they wish to view.
3. Click on "**Change Class**" button, and select "**Change_Request**", "**software**", or "**hardware**"	Selection will appear in Class Name field.
4. Click on local version of Class. Either "**GSFC_CHNG_REQ**" for "**Change_Request**", "**GSFC_NCR_SW**" for "**software**", or "**GSFC_NCR_HW**" for "**hardware**".	The selection will be highlighted indicating that only the local CCRs or NCRs will be viewed. We can not forward an ECS CCR or NCR.
5. Click on "**Select All**" button under the States column. Then De-Select '**D Duplicate**', '**P Defer**', and '**C Close**'.	This will select all the valid states for forwarding. This will allow the user to view the CCRs or NCRs as they transition states in DDTS.
6. Click on "**OK**" and return to main window.	Save selected class, project, and states for current session.
7. Select a previously submitted CCR or NCR from the DDTS main window listing.	The selected CCR or NCR will be displayed in the Record window.
8. Click on "**Change_State**" from the menu bar on the Record window. Select "**F Forward**".	In the upper left had corner of the CCR/NCR form, "**New Class**" and "**Forward To**" fields will be displayed.
9. Select the appropriate class, "**Change_Request**", "software" or "**hardware**".	
10. Select the appropriate project to "**Forward to**". If a **Change_Request** class, then "**ECS_CHNG_REQ**", if **software** class, "**ECS_NCR_SW**", or if **hardware** class, "**ECS_NCR_HW**".	If the user is forwarding a software or hardware class NCR, the form will prompt for a phone number. This is the phone number of the submitter.
11. Click on "**Commit**" button in DDTS main window.	C-MSS-40620 SMC-2515#A Submits state transition to database and forwards CCR/NCR to SMC database. Notice status field and Forwarding date information in CCR/NCR form.
12. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**FORWARDED**).
13. Double click on "**History Log**" and view transition information.	History log will report forwarding information (to, from) and the user that performed the action. Record bug ID for verification in SMC. (i.e. GSFcd00012)
	To verify CCR/NCR was forwarded to SMC:

14. Log into SMC workstation and execute DDTS. "**/data/CRM/ddts/bin/xddts**"	DDTS main window will appear in top right hand corner of window.
15. Click on "**Select**" from the DDTS main window menu bar. Click on "**Select Project, Class & State**".	The **Select Project, Class & State** dialog window will pop up. In this window the user will select the class, project, and states they wish to view.
16. Click on "**Change Class**" button, and select same class as selected from LSM session.	To view forwarded LSM CCR/NCR, must view same class as LSM.
17. Click on ECS related project. Either "**ECS_CHNG_REQ**" for **Change_Request** class, "**ECS_NCR_SW**" for **software** class, or "**ECS_NCR_HW**" for **hardware** class.	
18. Click on "**Clear**" button under the **States** column. Then select '**N New**' to view any new CCRs/NCRs.	Forwarded CCRs/NCRs are recorded as New submittals in the ECS projects.
19. Click on "**OK**" and return to main window.	Save selected class, project, and states for current session.
20. Find and select Forwarded CCR/NCR and view.	C-MSS-40650 SMC-2530#A (partial) Record the bug ID and note the forwarded date. Verify they are the same as the LSM version of the CCR/NCR.
21. Double click on the "**History**" log icon.	The History log will report forwarding information (from, to) and the user that performed the action. Verify all information against LSM history log.
	To verify distribution of information from SMC, view information at SMC from LSM.
22. Return to LSM, and change class and project to view SMC information. Click on "**Select**" from the DDTS main window menu bar. Click on "**Select Project, Class & State**".	The **Select Project, Class & State** dialog window will pop up. In this window the user will select the class, project, and states they wish to view.
23. Click on "**Change Class**" button, and select same class as selected from LSM session.	To view forwarded SMC CCR/NCR, must view same class as SMC.
24. Click on ECS related project. Either "**ECS_CHNG_REQ**" for **Change_Request** class, "**ECS_NCR_SW**" for **software** class, or "**ECS_NCR_HW**" for **hardware** class.	
25. Click on "**Clear**" button under the **States** column. Then select '**N New**' to view any new CCRs/NCRs.	Forwarded CCRs/NCRs are recorded as New submittals in the ECS projects.
26. Click on "**OK**" and return to main window.	Save selected class, project, and states for current session.
27. Find and select Forwarded CCR/NCR and view.	C-MSS-40660, C-MSS-40770 SMC-2530#A (partial), SMC-2535#A (partial), SMC-2540#A (partial) Record the bug ID and note the forwarded date. Verify they are the same as the LSM version of the CCR/NCR.
28. Exit.	

4.8.5.10 Test Case 10: Subscribe/View SMC Change Request Manager Test (TC024.0010)

This test case verifies the ability of a LSM CRM user to subscribe to a desired SMC CRM class and project and view the contents of a previously submitted non-conformance report (NCR) or configuration change request (CCR).

Test Configuration:

Hardware: LSM Workstation (msse3sun)

Software: Change Request Manager custom configuration

Data: previously submitted/forwarded NCR/CCR

Tools: DDTS

Test Input:

Inputs to this test case include DDTS commands and Change Request Manager configuration procedures.

Test Output:

The expected outputs of this test include the ability of the local Change Request Manager to subscribe and view contents of the SMC Change Request Manager.

Success Criteria:

This test is successful when the user is able to view the SMC Change Request Manager classes from the local Change Request Manager

Test Procedures:

Test Case ID: TC024.0010	
Test Name: Subscribe/View SMC CRM	
Test Steps:	Comments:
1. Log into LSM workstation and execute DDTS. "**/data/CRM/ddts/bin/xddts**"	DDTS main window will appear in top right hand corner of window.
2. Click on "**Select**" from the DDTS main window menu bar. Click on "**Select Project, Class & State**".	The **Select Project, Class & State** dialog window will pop up. In this window the user will select the class, project, and states they wish to view.
3. Click on "**Change Class**" button, and select "**Change_Request**", "**software**", or "**hardware**"	Selection will appear in Class Name field.
4. Click on SMC version of Class. Either "**ECS_CHNG_REQ**" for "**Change_Request**", "**ECS_NCR_SW**" for "**software**", or "**ECS_NCR_HW**" for "**hardware**".	The selection will be highlighted indicating that only the SMC CCRs or NCRs will be viewed.
5. Click on "**Select All**" button under the States column.	This will allow the user to view all CCRs or NCRs submitted within the SMC project.
6. Click on "**OK**" and return to main window.	Save selected class, project, and states for current session.

7. Select a previously submitted CCR or NCR from the DDTS main window listing.	C-MSS-40650, C-MSS-40660, C-MSS-40720, C-MSS-40760, C-MSS-40770 SMC-2520#A, SMC-2530#A, SMC-2535#A, SMC-2540#A The selected CCR or NCR will be displayed in the Record window. Notice the "**Remote Record - Modifications not allowed**" message at the bottom of the DDTS main window.
8. Double click on the "**History**" log in the Enclosures window.	The user is allowed to view the contents however, the user is not allowed to edit or delete the contents. (The "**Edit Enclosure**" and "**Delete Enclosure**" buttons are grayed out.)
9. Click on "**Next**" button in the Enclosure window.	The next enclosure will be displayed in the Enclosure window. Continue until all enclosures have been displayed.
10. Click on "**Dismiss**" button in bottom right corner of Enclosure window.	Enclosure window will disappear.

4.8.5.11 Test Case 11: Verify Required Field Entries Test (TC024.0011)

This test case verifies that all required field entries in the non-conformance report and the configuration change request forms assert entry information.

Test Configuration:

Hardware: LSM and SMC workstation

Software: Change Request Manager custom configuration

Data: N/A

Tools: DDTS

Test Input:

Inputs to this test case include valid and invalid field entries within both the NCR and CCR forms.

Test Output:

The expected outputs of this test include error messages if user does not enter information into required fields.

Success Criteria:

This test is successful when all required field entries acknowledge input and erroneous values.

Test Procedures:

Test Case ID: TC024.011	
Test Name: Verify Required Field Selections	
Test Steps:	
1. To verify this test, perform tests TC024.013 (Verify CRM User Screens), TC024.002 (Compose/Submit CCR), TC024.003 (Compose/Submit HW NCR), and TC024.004 (Compose/Submit SW NCR).	The verification of user screens will touch every field in the NCR and CCR forms. For verification of data entries, the submittal of NCR and CCR tests are performed.

4.8.5.12 Test Case 12: Verify "One of" Field Selection Test (TC024.012)

This test case verifies that all "one-of" field selections in the non-conformance report and the configuration change request forms contain specified attributes, and the fields do not allow the user to enter an invalid entry.

Test Configuration:

Hardware: LSM and SMC workstation (msse3sun, msse7sun)

Software: Change Request Manager custom configuration

Data: N/A

Tools: DDTS

Test Input:

Inputs to this test case include DDTS commands.

Test Output:

The expected outputs of this test include a valid selection from the one-of lists into the attribute field.

Success Criteria:

This test is successful when all "one-of" selection attributes acknowledge all input selections and no invalid entries..

Test Procedures:

Test Case ID: TC024.012	
Test Name: Verify "One of" Field Selection Test	
Test Steps:	Comments:
1. To verify this test, perform tests TC024.013 (Verify CRM User Screens), TC024.002 (Compose/Submit CCR), TC024.003 (Compose/Submit HW NCR), and TC024.004 (Compose/Submit SW NCR).	The verification of user screens will touch every field in the NCR and CCR forms. For verification of data entries, the submittal of NCR and CCR tests are performed.

4.8.5.13 Test Case 13: Verify Change Request Manager User Screens Test (TC024.013)

This test case verifies that the user entry screens for the configuration change request and non-conformance reports are consistent with documentation.

Test Configuration:

Hardware: LSM and SMC workstation

Software: Change Request Manager custom configuration

Data: N/A

Tools: DDTS

Test Input:

Inputs to this test case include DDTS commands and CCR and NCR field entries.

Test Output:

The expected outputs of this test include custom configured CCR and NCR user screens.

Success Criteria:

This test is successful when all CCR and NCR user screens conform to related documentation.

Test Procedures:

Test Case ID: TC024.013	
Test Name: Verify CRM User Screens	
Test Steps:	Comments:
1. Log into LSM workstation and execute DDTS. "**/data/CRM/ddts/bin/xddts**"	DDTS main window will appear in top right hand corner of window.
2. Click on "**Select**" from the DDTS main window menu bar. Click on "**Select Project, Class & State**".	The **Select Project, Class & State** dialog window will pop up. In this window the user will select the class, project, and states they wish to view.
3. Click on "**Change Class**" button, and select "**Change_Request**", "**software**", or "**hardware**"	Selection will appear in Class Name field.
4. Click on "**Select All**" button under the Projects column.	Both the **ECS_NCR_?W** and the **GSFC_NCR_?W** projects will be selected. (The **?** denotes either **S** for software or **H** for hardware.)
5. Click on "**Select All**" button under the States column.	This will select all the states. This will allow the user to view the NCRs as they transition states in DDTS.
6. Click on "**OK**" and return to main window.	Save selected class, project, and states for current session.
8. Click on "**Submit**" in DDTS main window.	Will bring up Submittal form.
9. Enter class to test. (i.e. Change_Request, software, or hardware.)	Class selection will determine project name to select.

10. Select project name. Enter "**?**" if unknown project names.	"**?**" will display project name help window with valid project names. Once entered project name, main submittal for will appear.
11. Transition through each entry field entering desired data type and size.	This verifies field lengths and data types according to documentation.
12. For each user entry screen, verify against documentation.	**Software** and **hardware** class submittal forms only have one page. **Change_Request** class contains 3 pages. To view each page of the Change_Request submittal form, user must transition through states.
13. Select "**Change State**" from menu bar and click on "**A Assign-Eval**"	
14. Transition through each entry field verifying documented data type and size.	
15. Select "**Change State**" from menu bar and click on "**O Assign-Implement**" or "**O Assign-Fix**"	Assign-Implement is for a CCR, Assign-Fix is for NCR
16. Transition through each entry field verifying documented data type and size.	
17. Select "**Change State**" from menu bar and click on "**R Implement**" or "**R Fix**"	Implement is for CCR, Fix is for NCR.
18. Transition through each entry field verifying documented data type and size.	
19. Select "**Change State**" from menu bar and click on "**T Assign-Verify**"	
20. Transition through each entry field verifying documented data type and size.	
21. Select "**Change State**" from menu bar and click on "**V Verify**"	Copy this record by clicking on "**Clone**" button in DDTS main window.
22. Transition through each entry field verifying documented data type and size.	
23. Select "**Change State**" from menu bar and click on "**C Close**"	
24. Transition through each entry field verifying documented data type and size.	
25. Select the cloned record. Select "**Change State**" from menu bar and click on "**D Duplicate**"	Cannot change the state of a Closed CCR/NCR
26. Transition through each entry field verifying documented data type and size.	
27. Select "**Change State**" from menu bar and click on "**F Forward**"	
28. Transition through each entry field verifying documented data type and size.	
29. When complete click on "**Commit**" in DDTs main window.	C-MSS-40670, C-MSS-40680, C-MSS-40700 SMC-2520#A Will submit form to database. Enclosures related to each NCR/CCR including **Problems**, **Proposed Change**, **Impact Summary**, **Resolution**, and **History** will be stored into the database with each NCR/CCR.

4.8.5.14 Test Case 14: Verify Report Production Test (TC024.014)

This test case verifies the ability of the Change Request Manager to produce reports based on Configuration Change Requests and Non-Conformance Reports. The Change Request Manager

contains three customized report formats, a full page report, a one line report, and a three line report.

Test Configuration:

Hardware: LSM and SMC workstation (msse3sun, msse7sun)

Software: Change Request Manager custom configuration

Data: previously submitted CCR or NCR

Tools: DDTS

Test Input:

Inputs to this test case include DDTS commands, and class/project designations.

Test Output:

The expected outputs of the test include viewable CRM reports with information corresponding to the desired output format.

Success Criteria:

This test is successful when the report outputs conform to report formats.

Test Procedures:

Test Case ID: TC024.014	
Test Name: Verify Report Production Test	
Test Steps:	Comments:
1. Log into workstation and execute DDTS. "**/data/CRM/ddts/bin/xddts**"	DDTS main window will appear in top right hand corner of window.
2. Click on "**Select**" from the DDTS main window menu bar. Click on **"Select Project, Class & State**".	The **Select Project, Class & State** dialog window will pop up. In this window the user will select the class, project and states they wish to view.
3. Click on "**Change Class**" button, and select "**Change_Request**", "**software**", or "**hardware**"	The selection will appear in the Class Name field.
4. Click on "**Select All**" button under the Projects column.	All projects related to selected class will be highlighted.
5. Click on "**Select All**" button under the **States** column.	This will select all the states. This will display all the associated CCRs or NCRs to the user.
6. Click on "**OK**" and return to main window.	Save selected class, project, and states for current session.
7. Select a previously submitted CCR or NCR from the DDTS main window listing.	The selected CCR or NCR will be displayed in the Record window.
8. Click on "**Print...**" button located in the lower middle portion of the DDTS main window.	A "**Printing Options**" window will appear.
	There are 3 custom report options, **Full Page Format**, **One Line Format**, **Three Line Format**

To print **FULL PAGE FORMAT**:	
9. Click on "**Print Full Page Format**" radio button, located in the top left portion of the **Printing Options** window	Selects the first of 3 custom reports. The "**Print Full Page Format**" text will become highlighted.
10. Click on "**Print Selected Bugs in Index**" radio button, located in the middle portion of the **Printing Options** window, under the section "**What to Print**"	The "**Print Selected Bugs in Index**" text will become highlighted. Only the selected CCR or NCR will be printed.
11. Display/Save/Print report.	Proceed to **PRINTING** section for printing options.
To print **ONE LINE FORMAT**:	
9. Click on "**Print One Line Format**" radio button, located in the top left portion of the **Printing Options** window, under the section "**Format to Print**".	Selects the second of the 3 three custom reports. The "**Print One Line Format**" text will become highlighted.
10. Click on "**Select Default Fields**" under the One Line Summary Field Selections column.	Will select and display "**identifier status severity headline**" as the fields to print.
11. Click on "**Print Selected Bugs in Index**" radio button, located in the middle portion of the **Printing Options** window, under the section "**What to Print**"	The "Print Selected Bugs in Index" text will become highlighted. Only the selected CCR or NCR will be printed.
12. Display/Save/Print report.	Proceed to **PRINTING** section for printing options.
To print **THREE LINE FORMAT**:	
9. Click on "**Print Three Line Format**" radio button, located in the top left portion of the **Printing Options** window	Selects the last of the 3 custom reports. The "**Print Three Line Format**" text will become highlighted.
10. Click on "**Print Selected Bugs in Index**" radio button, located in the middle portion of the **Printing Options** window, under the section "**What to Print**"	The "**Print Selected Bugs in Index**" text will become highlighted. Only the selected CCR or NCR will be printed.
11. Display/Save/Print report.	Proceed to **PRINTING** section for printing options.
PRINTING	There are 3 printing options: **Print to Process**, **Print to File**, **Print to Window**.
To **PRINT TO PROCESS**:	
13. Click on "**Print to Process**" radio button, locate in lower portion of the **Printing Options** window, under the section "**Where to Print**".	This option will print a report to the printer.
14. Enter "**/usr/bin/lp -d<printername>"** in the Name field.	This is the command used to print the report. Note, replace <printername> with destination printer name.
15. Click on "**Print...**" button in the **Printing Options** window.	The report will be sent to the destination printer.
To **PRINT TO FILE**:	
13. Click on "**Print to File**" radio button, located in lower portion of the **Printing Options** window, under the section "**Where to Print**".	This option will save a report to a file.
14. Enter a path and filename in the Filename field.	The report will be saved to this file.
15. Click on "**Print...**" button in the **Printing Options** window.	The report will be saved to the file.
To **PRINT TO WINDOW**:	
13. Click on "**Print to Window**" radio button, located in lower portion of the **Printing Options** window, under the section "**Where to Print**".	This option will display the report in a window.
14. Click on "**Print...**" button in the **Printing Options** window.	The report will be displayed in a window.

4.8.6 Management Services 1 Build (BC007)

The Management Services 1 build is a collection of the Performance Management Thread, the Hardware Fault Management Thread, the Management Data Access 1 Thread, the Accountability Thread, the Change Request Management Thread, and the Applications Management Services Build. The Applications Management Services Build is an aggregation of the External Interface Communications build, the Managed Process Framework thread, the MSS Lifecycle Services thread, the Discover/Monitor thread, and the MIB/Extensible Agent-Applications 2 thread. The External Interface Communications build is an aggregation of the User Account Management thread, the Kerberos ftp thread, the Kerberos Client/Server thread, and the Security Authorization thread.

The objective of this test is to verify that the previously tested functionality of the threads and builds remain intact following integration.

4.8.6.1 Test Case 1: Performance Management (BC007.001)

The Performance Management test is a regression test of the overall functionality of the Performance Management Thread. To ensure that the previously tested functionality of this thread remains intact upon integration with the Management Services 1 build.

This functionality includes the ability to:

- retrieve, monitor, display, and print the performance parameters, information of external, internal (site) network components, network component interfaces, all host, communication protocol stacks on demand and at configured intervals
- provide a configurable number of thresholds for each performance metric
- log, store, and retrieve all ECS performance statistics, summarized site status in/from the History Log
- extract and graph from the History Log the measured values for any performance metrics gathered for specified network objects
- receive, maintain and perform operational network performance benchmark test procedures
- display and print M&O staff-selected performance statistics, generate and redirect reports to the console, disk file, and printer

This test also verifies the performance issues related to the CSS and MSS services (operational availability, MDT), and the accuracy of network performance and network trending statistics being recorded in the database

<u>Test Configuration:</u>

Hardware: Network Analyzer, DCE Configured LAN

Software: MCI, Tivoli

Data: Configuration data, database activity

Tools: XRunner/LoadRunner

Test Input:

Inputs to this test case include keyins to set the degree of a fault and modify the time frame in which data will be gathered, Tivoli Network Node Manager commands, Normal network traffic on the system, available communication protocol stacks for monitoring, operational benchmark test procedures, increased volume of data distribution/access, XRunner and LoadRunner scripts to simulate multiple users performing multiple tasks, and configuration data.

Test Output:

Outputs to this test case include report performance information, comparison between received performance metrics and configured thresholds, writing/retrieving of selected performance management data to/from the History Log, summarized performance data at the local sites, extracting and graphing of selected performance management data for the selected time period. Also, the operational network performance benchmark test results, measurement and calculation of the operational availability and the MDT, performance statistics, current resource utilization information, and reports of the management data in a variety of outputs are obtained.

Success Criteria:

This test is successful when all of the designated network components, network hosts, and communication protocol stacks report the requested performance information in a timely manner, when all performance parameters exceeding their configured thresholds are flagged, when all selected performance management data has been properly written to and retrieved from the History Log, and graphed for the selected time interval. An error message should be displayed when a null interval is selected. All operational network performance benchmark tests pass under normal network conditions. If any error or warning conditions exist on the network, they should be flagged. This test should meet the required operational availability and MDT. Volume of data distribution/access can be increased successfully.

Test Procedures:

Test Case ID: BC007.001	
Test Name: Performance Management	
Test Steps:	Comments:
a. Performance Monitoring of Network Components	

1. For specified local and external network routers, request (on demand), display, and print the following: o operational status (on-line, off-line, or test mode) o type o speed o octets in/out o packets in/out o discards in/out o errors in/out	
2. Repeat the above for specified local and external network links.	
3. Repeat the above for specified local and external network bridges.	
4. Repeat the above for specified local and external network gateways.	
5. Utilize various configurable intervals for automatic performance monitoring, noting the above performance parameters for the aforementioned network components.	NI-0460#A
b. Performance Monitoring of Hosts	
6. For specified operating systems, request (on demand), display, and print the following: • total CPU utilization • memory utilization • physical disk I/Os • disk storage size • disk storage used • number of active processes • length of run queue • network I/0s (packets) • network errors	
7. Repeat above for specified local and external peripherals, also noting operational status (on-line, off-line, or test mode).	
8. Repeat above for specified local and external databases.	
9. Utilize various configurable intervals for automatic performance monitoring, noting the above performance parameters for the aforementioned hosts.	
c. Performance Monitoring of Communication Protocol Stacks	
10. For specified communication protocol stacks on internal (site) managed devices, request (on demand), display, and print the following: • number of transport layer messages received with errors • number of transport layer messages requiring retransmission • number of transport layer messages received that could not be delivered • number of network layer messages received with errors • number of network layer messages received that could not be delivered	

• number of network layer messages that were discarded	
11. Repeat above for specified communication protocol stacks on external devices.	
12. Utilize various configurable intervals for automatic performance monitoring, noting the above performance parameters for the aforementioned communication protocol stacks.	
d. History Log Verification	
13. Examine the History Log to ensure that all performance parameters which exceeded their configured thresholds have been recorded.	
14. Using APIs provided by CSS, ensure that the Performance Management Application Service can send selected ECS performance data to the History Log.	
15. Using APIs provided by CSS, ensure that the Performance Management Application Service can retrieve the following science algorithm performance data from the History Log: • algorithm name • algorithm version • start time • stop time • CPU utilization • memory utilization • disk reads • disk writes	
16. Using Tivoli Network Node Manager commands, extract summarized site information from logged performance data.	
17. Verify that the EMC PMAS can request and receive performance and summarized performance data from the site PMAS and other external systems.	
18. Verify that the site can send summarize data to the EMC PMAS.	
e. Performance Reporting	
19. Demonstrate the ability to graphically display the operational state of managed objects through HpOpenView.	
20. Demonstrate the ability of the PMAS to calculate statistics for: a. Mean Down Time (MDT), b. Mean Time Between Maintenance (MTBM); (Mean Time Between Preventive Maintenance (MTBPM) and Mean Time Between Corrective Maintenance (MTBCM)) and c. Mean Time to Repair (MTTR).	
21. Verify that the above statistics are stored in a repository accessible by the M&O Staff.	
22. Demonstrate, as an M&O staff user, the ability to display in tabular and graphical formats selected performance statistics.	
23. Print the statistics.	

24. Verify that the PMAS is capable of receiving system resource utilization information requests from SDPS Data Processing Subsystem, SDPS Data Server and Data Processing Subsystems and SDPS Client subsystem.	
25. Verify that the PMAS is capable of providing CPU utilization, memory utilization and disk I/O's to the SDPS Data Processing Subsystem, SDPS Data Server and Data Processing subsystems, and SDPS Client subsystem.	
26. Verify that the PMAS is capable of receiving resource utilization information requests from the SDPS subsystem.	
27. Verify that the PMAS has the ability to generate reports of the collected management data.	
28. Verify that these reports can be redirected to the console, a disk file and a printer.	DADS0901#A, PGS-0310#A, EOSD0510#A, EOSD0740#A, EOSD0750#A, EOSD0760#A, EOSD0500#A, EOSD1703#A, SMC-1345#A, SMC-3335, SMC-3345, SMC-3355, SMC-3375, SMC-3385, SMC-3395, SMC-3397, SMC-3415, EOSD0545, EOSD3490, EOSD3492, EOSD3620, EOSD3630, EOSD3700, EOSD3710, EOSD4035, EOSD4036, EOSD5000, EOSD5070, ESN-0815, ESN-1060, ESN-1065, ESN-1090, ESN-1206, ESN-1207, ESN-0240, ESN-0740, ESN-0750, ESN-0780, EOSD3510, EOSD3600, EOSD3625,

4.8.6.2 Test Case 2: Hardware Fault Management (BC007.002)

The Hardware Fault Management test is a regression test of the overall functionality of the Hardware Fault Management Thread which includes network device faults, operating system faults, and peripheral device faults. To ensure that the previously tested functionality of this thread remains intact upon integration with the Management Services 1 build.

This functionality includes the ability to:

- define fault categories
- assign faults to categories
- assign severity levels to faults
- detect a fault by polling an attribute of a managed object and by the receipt of a notification from another entity
- provide several mechanisms for the notification of a detected fault
- log all fault notifications for record keeping, report generation and post processing
- provide the required mechanisms to facilitate the proper diagnosis, isolation and identification of a fault

- provide minimum policies and procedures for fault management
- initiate the recovery procedures in order to restore the system to an operational state
- gather fault data, fault statistics (real-time and historical) in a graphical and tabular format via the MUI

Test Configuration:

Hardware: HP 9000/770

Software: HP-UX 9.05, HP OpenView B.04.00

Data: Managed hardware/software objects

Tools: DCE, Network Analyzer/Sniffer

Test Input:

Inputs to this test case include SNMP managed objects, different types of failures for proper detection by the FMAS, and utilities that perform diagnostics and testing for the isolated faults.

Test Output:

Expected outputs include alarms indicating fault detection, and display of faults on the system map and event log. An audit log of the error detection by the diagnostic tools is also created.

Success Criteria:

This test is successful if the following fault indications occur:

- Faults are detected by the Management Fault System and the correct alarms are generated
- Faults are displayed on the system map and the event logs. Audit logs are generated
- Diagnostic tools detect and log all errors

Test Procedures:

Test Case ID: BC007.002	
Test Name: Hardware Fault Management	
Test Steps:	Comments:
The following steps verify that the SNMP protocol is being used:	
1. Set up the Network Analyzer/Sniffer to monitor all traffic with the machine on which HP OpenView resides, ensuring that the filter is set to capture only SNMP traffic.	SNMP traffic should be captured.
The following steps verify the graphic topology:	
2. Log onto the designated FMAS machine. If remotely logged in, set the DISPLAY variable properly.	

3. Initialize HP OpenView by entering: **ovw &**	HP OpenView B.04.00 should come up and run in the background. EOSD1703#A
4. Verify that a map depiction of the network topology is accurately displayed.	
5. Verify that the lower level topologies include hosts, routers, communication lines, peripherals, and applications.	P3 - applications
6. Verify their status condition.	
7. Double-click on the available icons to verify that the lower level submaps exist and are accurately displayed.	
The following steps verify symbol manipulation:	
8. Using the "Edit" option, add a symbol to the map currently displayed.	
9. Using the "Edit" option, delete a different symbol from the map currently displayed.	
10. Click on a symbol on the map with the middle mouse button to drag it to a different location.	
11. Click on a symbol on the map with the right mouse button, then select the "Describe/Modify Symbol" option from the resulting pull-down menu.	The Symbol Description window will be displayed.
12. In the Symbol Description window, ensure that the "Status Source" is set to "Object".	This step is necessary to enable a change in symbol status (i.e., color).
13. From the Unix window where HP OpenView was initialized, enter the following for the object in Step 12: **/usr/OV/contrib/NNM/setStatus/setStatus** *<object> <new_status>*	The status/color of the object should change to the new status/color.
14. Click on a symbol on the map with the right mouse button, then select the "Describe/Modify Symbol" option from the resulting pull-down menu.	The Symbol Description window will be displayed.
15. In the Symbol Description window, ensure that the "Status Source" is set to "Symbol".	This step is necessary to enable a change in symbol type (i.e., shape).
16. Click on the symbol from Step 14 with the right mouse button, then select the "Change Symbol Type" option from the resulting pull-down menu.	The Change Symbol Type window will be displayed.
17. In the Change Symbol Type window, select a new symbol class and subclass, then click on the OK button.	The symbol type/shape of the object should change to the new type/shape.
The following steps verify the fault categories:	
18. Define a fault category via the Data Collection & Thresholds: SNMP window (Event Configurator sub-window).	
19. Assign specific faults to the defined category via the Data Collection & Thresholds: SNMP window (Event Configurator sub-window).	
20. Assign severity levels to each fault via the Data Collection & Thresholds: SNMP window (Event Configurator sub-window).	
The following steps verify the enabling/disabling of fault notifications:	NI-0430#A, NI-0440#A, NI-0450#A
21. Bring up the Event Configurator window via the Collection & Thresholds: SNMP window.	

22. Select the “Don’t log or display” option from the Event Category section.	ESN-0910#A
23. Force the selected event to occur.	Ensure that it has not been logged.
24. Return to the Event Configurator window, and select “Log only” from the Event Category section.	
25. Force the selected event to occur.	Ensure that it has been logged.
The following steps verify the capability of defining polling frequency.	
26. Modify the current polling rate for a MIB object via the Data Collection and Thresholds window.	Ensure that the polling rate has changed.
The following steps verify the display of on-line help windows:	
27. Examine the upper, right hand corner of all major HP OpenView windows.	Ensure that a Help pull-down menu label exists.
28. Select several menu items from the Help pull-down menu.	Ensure that the resultant Help windows contain information pertinent to its parent window. DADS0901#A, PGS-0310#A, EOSD0510#A, EOSD0740#A, EOSD0750#A, EOSD0760#A, EOSD0500#A, EOSD1703#A, SMC-4305, SMC-4315, SMC4325, SMC-4335, ESN-0810, ESN-0910, ESN-0920, ESN-1010, ESN-1030, ESN-0620, ESN-0690

4.8.6.3 Test Case 3: Management Data Access I (BC007.003)

The Management Data Access I test is a regression of the ECS Log Browser (ELB). To ensure that the previously tested functionality of the ELB remains intact following integration with other threads in the Management Services 1 build.

This test verifies the ability of the ECS Log Browser (ELB) to browse the management and application logs.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MDA User Interface, MDA Services

Data: Management log file, Application log file, XRunner script

Tools: MsMdPhase2 (test driver), XRunner 4.0

Test Input:

Inputs to this test case include keyins to open and read a record from a Management or Application log file.

Test Output:

Outputs to this test case include the description of event fields.

Success Criteria:

This test is successful when the Management or Application Log containing information on management events or ECS application can be browsed and the requested data is retrieved, displayed, and sorted.

Test Procedures:

Test Case ID: BC007.003	
Test Name: Management Data Access I	
Test Steps:	Comments:
1. On the test machine (SUN5), set the environment variable of FileDir with *log at the end: **setenv FileDir <***/usr/testa/kmc/*log***>**	This test cannot run on the HP machine in the mini-DAAC (msse4hp) with XRunner. At the present time, there isn't an HP license for XRunner.
2. Set the Display environment variable: **setenv DISPLAY** <*host ip address*>:0.0	
3.Change directory to where the test drivers reside: **cd** <*/usr/testa/kmc*>	
4. Execute the test program: **MsMdPhase2 &**	The MsMdPhase2 application will run in the background. The msse3sun terminal will be in the interactive mode. The ECS Log Browser is displayed on the screen.
5. From the command line: **ps**	This will verify that the MsMdPhase2 process is running.
6. Open another SUN5 session: **select xterm from the pulldown menu**	This session will be used for XRunner.
7. On the second SUN5 session (msse3sun): **setenv DISPLAY** <*host ip address*>:0.0	This sets the environment to display all functions to this window.
8. On the second SUN5 session: **xrun &**	This initiates XRunner.
9. From the XRunner window, **click Record and select Start Recording - Analog.**	The record function of XRunner is now running.
10.**Click the File pulldown menu.**	If the File pulldown menu is clicked, two choices of Open and Exit will appear.
11.**Click Open**.	If Open is clicked, the file selection box will appear giving the user the option to open a log file. Note: At the file selection box, select only the log files.
12.**Double click /usr/testa/kmc..**	Backs up to directory /usr/testa.
13.**Double click /usr/testa/..**	Backs up to directory /usr.
14.**Double click /usr/..**	
15.Scroll down to **/tmp** and double click.	
16.In the files area in the file Dialog Shell, there will be file ***/tmp/*.log***. Click on that file and **click OK**.	The log file opens in the ECS Browser Log window.
17.Select the first item (**Performance Poll**)and click OK.	The Event Field window appears. It will display three separate boxes: a) Host Information b) Event Field c) Performance Metric.
18.**Click close** to end Event Field window.	Event Field Window closes.

19.**Click Select pulldown menu for processing.**	When Select pulldown menu is clicked, three cascade pulldown menus of single select filters (by Category, Priority, Subsystem,) is displayed with the Custom Select option.
20.**Click 'Category'**	When 'Category' is selected, five cascade pulldown submenus are displayed: Fault, Performance, Security Alert, Topology Change, and Transaction. When one of the options is selected, all events of the selected option is displayed in the events box.
21.**Click 'Priority'**	When 'Priority' is selected, four cascade pulldown submenus are displayed: Low, Medium Low, Medium, and High. When one of the options is selected, all events of the selected option is displayed in the events box.
22.**Click 'Subsystem'**	When 'Subsystem' is selected, eight cascade pulldown submenus are displayed: CSS, DMS, DPS, DSS, INS, IOS, MSS, and PDPS. When one of the options is selected, all events of the selected option is displayed in the events box.
23.**From Select** pulldown menu, **click 'Custom Select'** or select the 'Custom Select' button at the bottom of the ECS Log Browser window.	When the 'Custom Select' is clicked, the custom select window is displayed. (Currently Category, Priority, Subsystem, and Type are available for testing. but ApplicationID, CSCI, InstanceID, Mode, ProcessID, and ProgramID functionally will be provided in Phase 3.
24.**Click cancel** to close window.	
25. **Click Sort** pulldown menu or Select pulldown menu for processing.	If Sort pulldown menu is clicked, six choices of single sort and a Custom Sort for multi-level sorting will appear: Time Stamp, Category, Type, Priority, Subsystem, CSCI.
26.**Click 'Time Stamp'**	All events will be listed in chronological order. Note: This process will take a few minutes to generate due to the size of the log.
27.**Click 'Category'**	All events will be grouped in categories. Note: This process will take a few minutes to generate due to the size of the log.
28.**Click 'Type'**	All events with same types will be grouped and listed. Note: This process will take a few minutes to generate due to the size of the log.
29.**Click 'Priority'**	All events with the same priorities will be grouped and listed. Note: This process will take a few minutes to generate due to the size of the log.
30.**Click 'Subsystem'**	All events will be grouped by susbsystems and listed. Note: This process will take a few minutes to generate due to the size of the log.
31.**Click 'CSCI'**	All events will be grouped by CSCI and listed. Note: This process will take a few minutes to generate due to the size of the log.
32.**From the Sort** pulldown menu, **click 'Custom Sort'** or click the 'Custom Sort' button at the bottom of the ECS Log Browser window.	When the Custom Sort is selected, the Custom Sort window appears and it gives the user the option to select multi-level of sorting in forward and reverse order of the following: Time Stamp, Category, Type, Priority, Subsystem, and CSCI.

33. **If Edit pulldown** menu is clicked, two choices of undo and reset will appear.	Undo = go back once reset = go back to the original (1st list)
34.**Select File - Exit**	The ECS Log Browser window closes.
35.On the **XRunner** window, **select Record - Stop Recording.**	This stops the XRunner animation.
36.**Select File - Save As**	This saves the animation to an XRunner script.
37.On the **XRunner** window, **select Open** and find the file that was just saved.	
38.**Select Replay - Animate**	XRunner script will then start processing.
39.**Select File - Exit to end XRunner.**	
40.End Test	

4.8.6.4 Test Case 4: Accountability (BC007.004)

The Accountability test is a regression test of the overall functionality of the Accountability Thread from phase 2 and 3. To ensure that the previously tested functionality of this thread remains intact upon integration with the Management Services 1 build. This test verifies the ability of the system to maintain accountability for multiple user registrations. Information and actions performed by each user are recorded and logged in an audit trail.

This test demonstrate the ability to:

- query user activity data, and data processing information stored in the Management Database
- report audit information to M&O staff via the MUI service
- retrieve user activity data (user id, type of user activity data items and date/time of activity)
- retrieve data processing information logged for specified data item, and from records generated
- generate reports from collected management data
- redirect reports to console, disk file and/or printer

Test Configuration:

Hardware: MSS Server

Software: MCI: MsAcUsrProfile, MsAcUsrReqMgr, MsAcGuestUserReg, EcAcOrderClient, EcAcRequestClient, MsAcOrderServer

Data: Sample user information, runserver.csh/runclient.csh drivers

Tools: DCE (rgy_edit), Sybase, Custom Test Drivers: runorderserver.csh, runorderclient.csh, runrequestclient.csh

Test Input:

Inputs to this test include user input information (user id, data items), placement of a data order and searching of logs, placement of a collected data management information.

Test Output:

Outputs to this test case include creation and insertion into the database, inputs to database regarding user activity data information, inputs to Management Data Access regarding specified user or data item, a printout (screen dump) of data processing information, queries based on test inputs and outputs of log data. A report that can be directed to a console, disk file and printer based on test inputs and outputs of log data.

Success Criteria:

This test is considered when the MSS Accountability Management database retrieves the information regarding user activity data generated by the Data Server, Data Processing and Client subsystems. The Management Data Access service can retrieve and query activities and database for data processing information associated with a particular user or data item stored in the database. All of the desired data information is generated to a report file for redirection to be send to a console, disk file and/or printer.

Test Procedures:

Test Case ID: BC007.004	
Test Name: Accountability	
Test Steps:	Comments:
a. Retrieve User Activity Data	
1. Login as a user capable of retrieving information from the Accountability Management Service.	
2. Set default to appropriate directory area.	
3. Display the Accountability menu page.	Accountability menu window appear.
4. Enter user id information	Access the Accountability database.
5. Select the Accountability button to retrieve user activity data information.	
6. Verify the retrieve screen is returned.	
7. Enter the activity data information to retrieve.	The activity selections are: user id, type of user activity, data items used, date/time activity.
8. Select the execute button to access the database for desired activity data. (browsed, searched, or ordered).	
9. Verify the activity data returned contains the appropriate information.	
b. Retrieve/Query Specified User/Data	
10. Select the Accountability button to retrieve information.	
11. Enter specified user or data item.	
12. Select the execute button to query the MDA for selected user or data item.	
13. Select to retrieve from the MDA service that particular information	
14. Verify the specified data information is returned.	
c. Retrieve/Query Data Process Information	
15. Select button for Data Processing Information.	
16. Retrieve the data processing information.	
17. Verify that the information could be retrieved.	

18. Retrieve all data processing information logged for a specific data item.	
19. Verify the specified data information is retrieved.	
20.Verify that the user can query the management database for all data processing information stored in the management database.	
d. Data Audit Trail	
21. Search the logs to find data audit information.	Stored local history logs.
22. Report audit information to user.	
23. Select to generate a report.	Verify report is generated.
24. Print the generated report.	
25. Select to direct report: a) console b) disk file c) printer	DADS0901#A, PGS-0310#A, EOSD0510#A, EOSD0740#A,EOSD0750#A,EOSD0760#A, EOSD0500#A, EOSD1703#A, ESN-0775#A, SMC-6315, SMC-6325, SMC-6335

4.8.6.5 Test Case 5: Change Request Manager (BC007.005)

The Change Request Manager test is a regression test of the overall functionality of the Change Request Manager Thread. To ensure that the previously tested functionality of this thread remains intact upon integration with the Management Services 1 build.

This test demonstrate the ability:

- of a user to compose and submit a Configuration Change Request (CCR) for system problems or enhancements, a hardware or software Non-Conformance Request (NCR)
- to transition a CCR or an NCR through the established states and assign the CCR or the NCR to a responsible engineer
- to forward a locally submitted NCR or CCR to the SMC Change Request Manager
- of a LSM CRM to subscribe and view contents of the SMC Change Request Manager
- of the Change Request Manager to produce reports

Test Configuration:

Hardware: LSM and SMC workstation

Software: Change Request Manager custom configuration

Data: N/A

Tools: DDTS

Test Input:

Inputs to this test case include DDTS commands and information required to generate and submit a CCR or an NCR.

Test Output:

The expected outputs of this test include a generated CCR or NCR which has been distributed as assigned (e-mailed to assigned users), an entry(s) in the history enclosure of modifications and the appearance of a new enclosure attached to a previously submitted CCR or NCR. Also, the outputs include the NCR/CCR forwarded to the SMC Change Request Manager and e-mail notification, viewable CRM reports with information corresponding to the desired output format.

Success Criteria:

This test is successful when a user is able to electronically draft a CCR/NCR, submit it, and track it through the approval process. Upon submittal, the user will receive an e-mail message containing the completed CCR/NCR form and its status within DDTS. Also, a user is able to transition a CCR/NCR, to recall a submitted CCR/NCR, to modify any of its contents, to forward local NCR/CCR to SMC, to view the SMC Change Request Manager classes from the local Change Request Manager. The report outputs should conform to report formats.

Test Procedures:

Test Case ID: BC007.005	
Test Name: Change Request Manager	
Test Steps:	Comments:
a. Compose/Submit a software NCR	
1. Log into workstation and execute DDTS. "**/data/CRM/ddts/bin/xddts**"	DDTS main window will appear in top right hand corner of window.
2. Click on "**Select**" from the DDTS main window menu bar. Click on "**Select Project, Class & State**".	The **Select Project, Class & State** dialog window will pop up. In this window the user will select the class, project, and states they wish to view.
3. Click on "**Change Class**" button, and select "**software**".	"**software**" will appear in Class Name field. This software class is associated with the associated site Change Request Manager.
4. Click on desired project, either "**ECS_NCR_SW**" or "**GSFC_NCR_SW**".	The chosen project will become highlighted. If the "**Select All**" button were pressed, both the **ECS_NCR_SW** and/or the **GSFC_NCR_SW** would be selected.
5. Click on "**Select All**" button under the **States** column.	This will select all the states. This will allow the user to view the NCRs as they transition states in DDTS.
6. Click on "**OK**" and return to main window.	Save selected class, project, and states for current session.
7. Click on "**Submit**" button to begin.	The "**Submit a New Defect**" form will be displayed in the Record window.
8. Enter "**software**" at the class prompt.	
9. Enter "**GSFC_NCR_SW**" or "**ECS_NCR_SW**" at project name prompt.	Loads **GSFC_NCR_SW** or **ECS_NCR_SW** form into Record window.
10. At each prompt enter information based on category. Enter a valid entry for each "one of" or multiple entry fields. Also enter an invalid entry in each of the "one of" or multiple entry fields.	Since the data fields are of data type character, any character is a valid entry. The "one of" fields and multiple entry fields will not allow invalid entries.
11. When complete, click on "**Commit**" button in DDTS main window.	The commits the NCR into the database.

12. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**New**).
b. State Transition of an NCR	
13. Click on "**Select**" from the DDTS main window menu bar. Click on "**Select Project, Class & State**".	The **Select Project, Class & State** dialog window will pop up. In this window the user will select the class, project, and states they wish to view.
14. Click on "**Change Class**" button, and select "**software**" or "**hardware**".	User selection will appear in the Class Name field. These NCRs are associated with the associated site Change Request Manager.
15. Click on desired project, either "**ECS_NCR_?W**" or "**GSFC_NCR_?W**".	The chosen project will become highlighted. If the "**Select All**" button were pressed, both projects would be selected. NOTE, the "**?**" denotes either an "**S**" or "**H**" depending on the class selected.
16. Click on "**Select All**" button under the **States** column.	This will select all the states. This will allow the user to view the NCRs as they transition states in DDTS.
17. Click on "**OK**" and return to main window.	Save selected class, project, and states for current session.
18. Select a previously submitted NCR from the DDTS main window listing.	The selected NCR will be displayed in the Record window.
If transitioning **SOFTWARE** class:	
19. Click on "**Change_State**" from the menu bar on the Record window. Select "**A Assign-Eval**".	The "**ANALYSIS INFORMATION**" column will appear in the NCR form.
20. Enter information related to the specified prompts.	
21. Click on the "**Commit**" button in the DDTS main window to store the state transition to the database.	Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **ASSIGNED-EVAL**.
22. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**ASSIGNED-EVAL**).
23. Click on "**Change_State**" from the menu bar on the Record window. Select "**O Assign-Fix**".	Additional information will appear below the "**ANALYSIS INFORMATION**" column.
24. Enter information related to the specified prompts.	Note, the "**Affects Subsystem**", "**CSCI**", "**Problem Type**", "**Recommended Change**", and "**Phase Problem Caused**" are all one-of fields. Test each option for completion.
25. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **ASSIGNED_FIX**.
26. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**ASSIGNED-FIX**).
27, Click on "**Change_State**" from the menu bar on the Record window. Select "**R Fix**".	The NCR will transition to page 2, and the "**RESOLUTION INFORMATION**" column will appear.
28. Enter information related to the specified prompts.	Note, "**Resolution**", and "**Phase Problem Fixed**" attributes are one-of fields. Test each option for completion.

29. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **FIXED**.
30. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**FIXED**).
31. Click on "**Change_State**" from the menu bar on the Record window. Select "**T Assign-Verify**".	The NCR will transition to page 2, and the "**TESTING INFORMATION**" column will appear.
32. Enter information related to the specified prompts.	
33. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **ASSIGN-VERIFY**.
34. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**ASSIGNED-VERIFY**).
35. Click on "**Change_State**" from the menu bar on the Record window. Select "**V Verify**".	The NCR will transition to page 2, and the "**VERIFICATION INFORMATION**" column will appear.
36. Enter '**Test Status**' information.	The '**Test Status**' field has 2 choices, Pass, Fail, for a total of 2 tests.
37. If the user selects to add an Enclosure, click on "**Modify**" from menu bar on the Record window, and select "**Add Enclosure**".	An Enclosure window will appear. Enter information.
38. When complete, click on "**File**" from the menu bar on the Editor window, and select "**Save Changes & Dismiss Editor**".	Saves Enclosure and prompts for title of enclosure.
39. Enter the name of the enclosure and click on "**OK**".	The enclosure will appear in the Enclosures window.
40. At '**Enclosure Added**' prompt enter Y or N.	Notice that the status field on page 1 of the CCR form has changed to '**Verified'**
41. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **VERIFIED**.
42. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**VERIFIED**).
43. To **CLOSE** an NCR, Click on "**Change_State**" from the menu bar on the Record window. Select "**C Close**".	The NCR will transition to page 2, and the "**CLOSING INFORMATION**" column will appear.
44. Enter information related to the specified prompts.	
45. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **CLOSED**.
46. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**CLOSED**).
47. Double click on the "**History**" icon in the Enclosures window.	View the contents of the enclosure and verify state transition and user information for validity.

48. To **DUPLICATE** an NCR, Click on "**Change_State**" from the menu bar on the Record window. Select "**D Duplicate**".	If, "**ANALYSIS INFORMATION**" has not been entered, Enter information now. Note, cannot duplicate a CLOSED NCR.
49. Enter an NCR# which this NCR is a duplicate of.	If an invalid NCR# is entered, DDTS will inform that the NCR# does not exist and requests valid NCR# be entered.
50. Click on "**Commit**" button in the DDTS main window to store the state transition to the database.	Stores the transition of the NCR to the database, and updates the DDTS main window to reflect the state change. Notice the Status field indicates **DUPLICATE**.
51. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**DUPLICATED**).
52. Double click on the "**History**" icon in the Enclosures window.	View the contents of the enclosure and verify state transition and user information for validity.
c. Forward Local NCR/CCR to SMC	
53. Click on "**Select**" from the DDTS main window menu bar. Click on "**Select Project, Class & State**".	The **Select Project, Class & State** dialog window will pop up. In this window the user will select the class, project, and states they wish to view.
54. Click on "**Change Class**" button, and select "**Change_Request**", "**software**", or "**hardware**"	Selection will appear in Class Name field.
55. Click on local version of Class. Either "**GSFC_CHNG_REQ**" for "**Change_Request**", "**GSFC_NCR_SW**" for "**software**", or "**GSFC_NCR_HW**" for "**hardware**".	The selection will be highlighted indicating that only the local CCRs or NCRs will be viewed. We can not forward an ECS CCR or NCR.
56. Click on "**Select All**" button under the States column. Then De-Select '**D Duplicate**', '**P Defer**', and '**C Close**'.	This will select all the valid states for forwarding. This will allow the user to view the CCRs or NCRs as they transition states in DDTS.
57. Click on "**OK**" and return to main window.	Save selected class, project, and states for current session.
58. Select a previously submitted CCR or NCR from the DDTS main window listing.	The selected CCR or NCR will be displayed in the Record window.
59. Click on "**Change_State**" from the menu bar on the Record window. Select "**F Forward**".	In the upper left had corner of the CCR/NCR form, "**New Class**" and "**Forward To**" fields will be displayed.
60. Select the appropriate class, "**Change_Request**", "software" or "**hardware**".	
61. Select the appropriate project to "**Forward to**". If a **Change_Request** class, then "**ECS_CHNG_REQ**", if **software** class, "**ECS_NCR_SW**", or if **hardware** class, "**ECS_NCR_HW**".	If the user is forwarding a software or hardware class NCR, the form will prompt for a phone number. This is the phone number of the submitter.
62. Click on "**Commit**" button in DDTS main window.	Submits state transition to database and forwards CCR/NCR to SMC database. Notice status field and Forwarding date information in CCR/NCR form.
63. Check e-mail for incoming messages from DDTS..	User will receive an e-mail from DDTS informing them about the NCR that was submitted, and its current state (**FORWARDED**).
64. Double click on "**History Log**" and view transition information.	History log will report forwarding information (to, from) and the user that performed the action. Record bug ID for verification in SMC. (i.e. GSFcd00012)
	To verify CCR/NCR was forwarded to SMC:

65. Log into SMC workstation and execute D5TS. "**/data/CRM/ddts/bin/xddts**"	DDTS main window will appear in top right hand corner of window.
66. Click on "**Select**" from the DDTS main window menu bar. Click on "**Select Project, Class & State**".	The **Select Project, Class & State** dialog window will pop up. In this window the user will select the class, project, and states they wish to view.
67. Click on "**Change Class**" button, and select same class as selected from LSM session.	To view forwarded LSM CCR/NCR, must view same class as LSM.
68. Click on ECS related project. Either "**ECS_CHNG_REQ**" for **Change_Request** class, "**ECS_NCR_SW**" for **software** class, or "**ECS_NCR_HW**" for **hardware** class.	
69. Click on "**Clear**" button under the **States** column. Then select '**N New**' to view any new CCRs/NCRs.	Forwarded CCRs/NCRs are recorded as New submittals in the ECS projects.
70. Click on "**OK**" and return to main window.	Save selected class, project, and states for current session.
71. Find and select Forwarded CCR/NCR and view.	Record the bug ID and note the forwarded date. Verify they are the same as the LSM version of the CCR/NCR.
72. Double click on the "**History**" log icon.	The History log will report forwarding information (from, to) and the user that performed the action. Verify all information against LSM history log.
d. Subscribe/View SMC CRM	
73. Click on "**Select**" from the DDTS main window menu bar. Click on "**Select Project, Class & State**".	The **Select Project, Class & State** dialog window will pop up. In this window the user will select the class, project, and states they wish to view.
74. Click on "**Change Class**" button, and select "**Change_Request**", "**software**", or "**hardware**"	Selection will appear in Class Name field.
75. Click on SMC version of Class. Either "**ECS_CHNG_REQ**" for "**Change_Request**", "**ECS_NCR_SW**" for "**software**", or "**ECS_NCR_HW**" for "**hardware**".	The selection will be highlighted indicating that only the SMC CCRs or NCRs will be viewed.
76. Click on "**Select All**" button under the States column.	This will allow the user to view all CCRs or NCRs submitted within the SMC project.
77. Click on "**OK**" and return to main window.	Save selected class, project, and states for current session.
78. Select a previously submitted CCR or NCR from the DDTS main window listing.	The selected CCR or NCR will be displayed in the Record window. Notice the "**Remote Record - Modifications not allowed**" message at the bottom of the DDTS main window.
79. Double click on the "**History**" log in the Enclosures window.	The user is allowed to view the contents however, the user is not allowed to edit or delete the contents. (The "**Edit Enclosure**" and "**Delete Enclosure**" buttons are grayed out.)
80. Click on "**Next**" button in the Enclosure window.	The next enclosure will be displayed in the Enclosure window. Continue until all enclosures have been displayed.
81. Click on "**Dismiss**" button in bottom right corner of Enclosure window.	Enclosure window will disappear.
e. Verify Report Production Test	
82. Select a previously submitted CCR or NCR from the DDTS main window listing.	The selected CCR or NCR will be displayed in the Record window.
83. Click on "**Print...**" button located in the lower middle portion of the DDTS main window.	A "**Printing Options**" window will appear.

	There are 3 custom report options, **Full Page Format**, **One Line Format**, **Three Line Format**
To print **FULL PAGE FORMAT**:	
84. Click on "**Print Full Page Format**" radio button, located in the top left portion of the **Printing Options** window	Selects the first of 3 custom reports. The "**Print Full Page Format**" text will become highlighted.
85. Click on "**Print Selected Bugs in Index**" radio button, located in the middle portion of the **Printing Options** window, under the section "**What to Print**"	The "**Print Selected Bugs in Index**" text will become highlighted. Only the selected CCR or NCR will be printed.
86. Display/Save/Print report.	Proceed to **PRINTING** section for printing options.
To print **ONE LINE FORMAT**:	
87. Click on "**Print One Line Format**" radio button, located in the top left portion of the **Printing Options** window, under the section "**Format to Print**".	Selects the second of the 3 three custom reports. The "**Print One Line Format**" text will become highlighted.
88. Click on "**Select Default Fields**" under the One Line Summary Field Selections column.	Will select and display "**identifier status severity headline**" as the fields to print.
89. Click on "**Print Selected Bugs in Index**" radio button, located in the middle portion of the **Printing Options** window, under the section "**What to Print**"	The "Print Selected Bugs in Index" text will become highlighted. Only the selected CCR or NCR will be printed.
90. Display/Save/Print report.	Proceed to **PRINTING** section for printing options.
To print **THREE LINE FORMAT**:	
91. Click on "**Print Three Line Format**" radio button, located in the top left portion of the **Printing Options** window	Selects the last of the 3 custom reports. The "**Print Three Line Format**" text will become highlighted.
92. Click on "**Print Selected Bugs in Index**" radio button, located in the middle portion of the **Printing Options** window, under the section "**What to Print**"	The "**Print Selected Bugs in Index**" text will become highlighted. Only the selected CCR or NCR will be printed.
93. Display/Save/Print report.	Proceed to **PRINTING** section for printing options.
PRINTING	There are 3 printing options: **Print to Process**, **Print to File**, **Print to Window**.
To **PRINT TO PROCESS**:	
94. Click on "**Print to Process**" radio button, locate in lower portion of the **Printing Options** window, under the section "**Where to Print**".	This option will print a report to the printer.
95. Enter "**/usr/bin/lp -d<printername>"** in the Name field.	This is the command used to print the report. Note, replace <printername> with destination printer name.
96. Click on "**Print…**" button in the **Printing Options** window.	The report will be sent to the destination printer.
To **PRINT TO FILE**:	
97. Click on "**Print to File**" radio button, located in lower portion of the **Printing Options** window, under the section "**Where to Print**".	This option will save a report to a file.
98. Enter a path and filename in the Filename field.	The report will be saved to this file.
99. Click on "**Print…**" button in the **Printing Options** window.	The report will be saved to the file.
To **PRINT TO WINDOW**:	

100. Click on "**Print to Window**" radio button, located in lower portion of the **Printing Options** window, under the section "**Where to Print**".	This option will display the report in a window.
101. Click on "**Print...**" button in the **Printing Options** window.	The report will be displayed in a window. DADS0901#A, PGS-0310#A, EOSD0510#A, EOSD0740#A, EOSD0750#A, EOSD0760#A, EOSD0500#A, EOSD1703#A

4.9 Communications Infrastructure 2 Tests

The following threads and builds verify the functionality of message passing and Multicasting in a basic communications environment.

- Message Passing 2 Thread (TC025)
- Communication Infrastructure 2 Build (BC008)

4.9.1 Message Passing Two Thread (TC025)

The purpose of this thread is to verify the functionality of implementation two of the Message Passing Service. The Message Passing Two API consists of a class containing several virtual functions that developers will override in order to implement application specific handling and processing of messages. The API will spawn a thread which will send the message according to the overridden virtual functions defined by the programmer. This implementation provides mechanisms for developers to utilize both asynchronous and deferred synchronous communication in their applications. These methods of communication will be utilized in a multi-platform DCE environment consisting of multiple instances of various client and server applications.

Asynchronous message passing mode occurs when the client makes a non-blocking request to the server; i.e. without loosing interactive control. The request(call) will not return anything and is just used to pass data to a server. This allows client execution to continue while the server processes the received data.

Deferred synchronous message passing mode occurs when the client makes a non-blocking request(call) and receives an acknowledgment of the call, but no immediate results. The client is free to continue executing while the server processes the client request. The results are then stored in an intermediate buffer (the message object) where they can be retrieved at a later time by the client.

4.9.1.1 Test Case 1: Asynchronous Message Passing (TC025.001)

The purpose of this test is to verify that implementation two of the Message Passing Service has the capability to transmit messages/data via the asynchronous message passage mode of communication. Specifically, this test demonstrates that implementation two of the Message Passing Service:

- provides a non blocking API for clients(senders) to send messages to servers(receivers) asynchronously.
- can associate the message object to a return value that is available upon request to the sender.
- provides an API for senders to obtain return information concerning the transmission of the message.

Test Configuration:

Hardware: HP and Sun workstations/servers HCI, Internetworking subsystem CI (LAN Network resources)-INCI

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions

Data: N/A

Tools: cli_hp,srv_hp,cli_sun,srv_sun.

Test Input:

The name under which the test server application (e.g. srv_sun) will be registered in the cell directory service must be supplied to both the test server and client applications.

Test Output:

The client test driver should issue statements indicating that a message was sent and it is continuing its execution. It should also display the uuid of the message object, and the return status of the message transmission. The server test driver should issue a statement indicating that the message was received.

Success Criteria:

The client application sends a message to the server and continues executing and subsequently retrieves return information indicating that the message thread executed successfully.

Test Case ID: TC025.001	
Test Name: Asynchronous Messaging Passing	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.
2. Create two sessions (i.e. open two xterm windows) and log into DCE in both by typing : **dce_login** *username password*	

3. In one window, start the cdsbrowser application in the background by typing: **cdsbrowser &**	Note the cdsbrowser application is available only on HP platforms. When executing this testcase on other platforms create a third session in step 2. In this window log into a HP machine in the same cell as the test machine, set your DISPLAY variable to point back to the test machine, and then execute step 3.
4. Start the test server application by typing the following in one of the test machine windows: **srv_hp** *<cds entry name>*	The cds entry should be placed in a directory in which the executor of the test has write permission. For example, in the relacell the following is a valid entry name: */.:/RelA/msg_server* where */.:/RelA/* is the directory and *msg_server* is the server name. A message stating "The server is now listening...." should appear in the server window indicating that the server is running.
5. Verify the server object has been registered in the CDS by clicking the **tree->update** button in the cdsbrowser's toolbar.	The server object should now be displayed inside the user-specified directory (e.g. underneath the */.:/RelA/* directory)
6. In the second window start the client application by typing the following: **cli_hp** *<cds entry name>*	The cds entry name should correspond to the cds entry name specified in step 4.
7. Verify the following message appears in the client window indicating the message was issued: "Sending message to *receiver...*"	The message is sent using the EcDcDSyncCom::Send method. (C-CSS-22000)
8. Verify the following message appears in the client window indicating the client has regained control of the process's execution: "Continuing execution..."	This message is displayed immediately after the EcDcDSyncCom::Send method is invoked. It should appear prior to the message displayed by the server in the next step. (C-CSS-22000) The client test driver will now sleep until the message execution is complete. This is required so the message in step 11 will be as indicated.
9. Verify the following message appears in the server window indicating the server has received the message: "Message received from client..."	
10. Verify the following message appears in the client window indicating that the client has obtained an UUID corresponding to the message object: "Message UUID is: *<returned uuid>*"	This UUID corresponds to the message object being transmitted. (C-CSS-22130)
11. Verify the following message appears in the client window indicating the return status of the message corresponds to successful completion: "Message return status is: 1"	The message/thread return status is stored in the private member "_done" and can be obtained using the message object's Done() member function. A value of 1 indicates the thread completed successfully and -1 indicates the thread failed in some point of its execution. If the thread isn't complete a value of 0 will be returned. (C-CSS-22140)
12. Kill the server process by issuing a kill signal in the server window. (e.g. **Ctrl-C)**	
13. Remove the server object from the cds namespace by highlighting the server object in the cdsbrowser and clicking on the **options->delete** entry button in the toolbar.	

14. In each window logout of dce by issuing the following commands: **kdestroy** **exit**	

4.9.1.2 Test Case 2: Deferred Synchronous Message Passing (TC025.002)

The purpose of this test is to verify that implementation two of the Message Passing Service has the capability to transmit messages/data via the Deferred Synchronous message passing mode of communication. Specifically, this test demonstrates that implementation two of the Message Passing Service:

- provides a non blocking API for clients(senders) to send messages to servers(receivers) in a deferred synchronous manner.
- stores results returned by the server(receiver) in an intermediary which the client may contact at a later time in order to retrieve the results.
- can associate the message object to a return value that is available upon request to the sender.
- provides an API for senders to obtain return information concerning the transmission of the message.

Test Configuration:

Hardware: HP and Sun workstations/servers HCI, Inter networking subsystem CI (LAN Network resources), INCI

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions

Data: Tester supplied input

Tools: def_cli_hp,def_srv_hp,def_cli_sun,def_srv_sun.

Test Input:

The name under which the test server application (e.g. def_srv_sun) will be registered in the cell directory service must be supplied to both the test server and client applications. The client application will also prompt the user for two numbers to be multiplied. .

Test Output:

The client test driver should issue statements indicating that a message was sent and it is continuing its execution. It should also display the uuid of the message object, the return status of the message transmission, and the result returned by the server. The server test driver should issue a statement indicating that the message was received.

Success Criteria:

The client application sends a message to the server and continues executing and subsequently retrieves return and result information indicating that the message thread executed successfully.

Test Case ID: TC025.002	
Test Name: Deferred Synchronous Messaging Passing	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.
2. Create two sessions (i.e. open two xterm windows) and log into DCE in both by typing : **dce_login** *username password*	
3. In one window, start the cdsbrowser application in the background by typing: **cdsbrowser &**	Note the cdsbrowser application is available only on HP platforms. When executing this testcase on other platforms create a third session in step 2. In this window log into a HP machine in the same cell as the test machine, set your DISPLAY variable to point back to the test machine, and then execute step 3.
4. Start the test server application by typing the following in one of the test machine windows: **def_srv_hp** *<cds entry name>*	The cds entry should be placed in a directory in which the executor of the test has write permission. For example, in the relacell the following is a valid entry name: */.:/RelA/msg_server* where */.:/RelA/* is the directory and *msg_server* is the server name. A message stating "The server is now listening...." should appear in the server window indicating that the server is running.
5. Verify the server object has been registered in the CDS by clicking the **tree->update** button in the cdsbrowser's toolbar.	The server object should now be displayed inside the user-specified directory (e.g. underneath the */.:/RelA/* directory)
6. In the second window start the client application by typing the following: **def_cli_hp** *<cds entry name>*	The cds entry name should correspond to the cds entry name specified in step 4.
7. When prompted by the client test driver, enter the first number to be multiplied and press **Enter**.	
8. When prompted by the client test driver, enter the second number to be multiplied and press **Enter**.	
9. Verify the following message appears in the client window indicating the message was issued: "Sending message *<message #>* to *receiver...*"	The message is sent using the EcDcDSyncCom::Send method. (C-CSS-22010)

10. Verify the following message appears in the client window indicating the client has regained control of the process's execution: "Continuing execution..."	This message is displayed immediately after the EcDcDSyncCom::Send method is invoked. It should appear prior to the message displayed by the server in the next step. (C-CSS-22010) The client test driver will now sleep until the message execution is complete. This is required so the message in step 13 will be as indicated.
11. Verify the following message appears in the server window indicating the server has received the message: "Message received from client..."	
12. Verify the following message appears in the client window indicating that the client has obtained an UUID corresponding to the message object: "Message UUID is: *<returned uuid>*"	This UUID corresponds to the message object being transmitted. (C-CSS-22130)
13. Verify the following message appears in the client window indicating the return status of the message corresponds to successful completion: "Message return status is: 1"	The message/thread return status is stored in the private member "_done" and can be obtained using the message object's Done() member function. A value of 1 indicates the thread completed successfully and -1 indicates the thread failed in some point of its execution. If the thread isn't complete a value of 0 will be returned. (C-CSS-22140)
14. Verify the following message appears in the client window indicating the result returned from the server routine. "Result is *<appropriate result>*"	In this step the client application retrieves the result from the message object via the object's GetResults() method. The client then displays the result on the screen. (C-CSS-22010)
15. Kill the server process by issuing a kill signal in the server window. (e.g. **Ctrl-C)**	
16. Remove the server object from the cds namespace by highlighting the server object in the cdsbrowser and clicking on the **options->delete** entry button in the toolbar.	
17. In each window logout of dce by issuing the following commands: **kdestroy** **exit**	

4.9.1.3 Test Case 3: Retransmission of Unsuccessful Message (TC025.003)

The purpose of this test is to verify that implementation two of the Message Passing Service can defer sending a message to a receiver when the receiver is not active. Specifically, this test will demonstrate the Message Passing Service's ability to retry sending a message a specified number of times at specified time intervals if the initial transmission is unsuccessful.

Test Configuration:

Hardware: HP and Sun workstations/server HCI, Inter networking subsystem CI (LAN Network resources), INCI

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions

Data: N/A

Tools: ret_cli_hp,srv_hp,ret_cli_sun,srv_sun.

Test Input:

The name under which the test server application (e.g. ret_srv_sun) will be registered in the cell directory service must be supplied to both the test server and client applications. The client application will also prompt the user for the number of times to try sending the message as well as the time interval between transmission attempts.

Test Output:

The client test driver should issue statements indicating that a message was sent and it is continuing its execution. Subsequently, messages indicating how many transmissions have been attempted and the system time of the current attempt are displayed. When the message transmission is completed the uuid of the message object, and the return status of the message transmission will be displayed. The server test driver should issue a statement indicating that the message was received.

Success Criteria:

The client application sends a message to the server and continues executing. The Message Service retransmits the message until it is successful in connecting to the server. Upon completion of the message transmission the client retrieves the return information indicating that the message thread executed successfully.

Test Case ID: TC025.003	
Test Name: Retransmission of Unsuccessful Message	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.
2. Create two sessions (i.e. open two xterm windows) and log into DCE in both by typing : **dce_login** *username password*	
3. In one window, start the cdsbrowser application in the background by typing: **cdsbrowser &**	Note the cdsbrowser application is available only on HP platforms. When executing this testcase on other platforms create a third session in step 2. In this window log into a HP machine in the same cell as the test machine, set your DISPLAY variable to point back to the test machine, and then execute step 3.
4. Start the test client application by typing the following in one of the test machine windows: **ret_cli_hp** *<cds entry name>*	The cds entry name should correspond to the cds entry name that will be used in registering the server application. See step11 for more details.
5. When prompted by the client test driver, enter the number of times to try sending the message and press **Enter**.	Enter a number large enough so that there will be time to start the server application in the second window before the Message Service stops trying to retransmit the message.

6. When prompted by the client test driver, enter the time interval to wait between transmission attempts and press **Enter**.	The time interval should be in accordance with the number of retransmission attempts so that the server application can be started before the Message Service stops trying to retransmit the message.
7. Verify the following message appears in the client window indicating the message was issued: "Sending message to *receiver...*"	The message is sent using the EcDcDSyncCom::Send method. (C-CSS-22000)
8. Verify the following message appears in the client window indicating the client has regained control of the process's execution: "Continuing execution..."	This message is displayed immediately after the EcDcDSyncCom::Send method is invoked. (C-CSS-22000) The client test driver will now sleep until the message is transmitted successfully. This is required so the message in step 15 will be as indicated.
9. Verify the following message appears in the client window indicating the message thread is in its initial attempt at execution: "Current transmission attempt: 1." "Time of attempt:*<current time>*"	Messages of this format will continue to appear in the client window until the message is transmitted successfully.(C-CSS-22150)
10. Verify the following message appears in the client window indicating the message thread could not locate the server object: "**DCE Exception: entry not found (dce/rpc)"	This DCE Exception occurs because the server application has not been registered in the cell's namespace. A message of this type will occur every time a transmission attempt fails because the object could not be found.
11. In the second window start the test server application by typing the following: **srv_hp** *<cds entry name>*	The cds entry should be placed in a directory in which the executor of the test has write permission. For example, in the relacell the following is a valid entry name: */.:/RelA/msg_server* where */.:/RelA/* is the directory and *msg_server* is the server name. A message stating "The server is now listening...." should appear in the server window indicating that the server is running.
12. Verify the server object has been registered in the CDS by clicking the **tree->update** button in the cdsbrowser's toolbar.	The server object should now be displayed inside the user-specified directory (e.g. underneath the */.:/RelA/* directory)
13. Verify the following message appears in the server window indicating the server has received the message: "Message received from client..."	This message should appear on the client's first attempt to retransmit the message after step 11 is completed. However, because the server's registration is not instantaneous there is a very minute chance that a retransmission attempt occurring almost simultaneously with the server's instantiation will fail. Under no circumstances, however, should the subsequent transmission attempt fail.
14. Verify the following message appears in the client window indicating that the client has obtained an UUID corresponding to the message object: "Message UUID is: *<returned uuid>*"	This UUID corresponds to the message object being transmitted. (C-CSS-22130)
15. Verify the following message appears in the client window indicating the return status of the message corresponds to successful completion: "Message return status is: 1"	The message/thread return status is stored in the private member "_done" and can be obtained using the message object's Done() member function. A value of 1 indicates the thread completed successfully and -1 indicates the thread failed in some point of its execution. If the thread isn't complete a value of 0 will be returned. (C-CSS-22140)

16. Kill the server process by issuing a kill signal in the server window. (e.g. **Ctrl-C)**	
17. Remove the server object from the cds namespace by highlighting the server object in the cdsbrowser and clicking on the **options->delete** entry button in the toolbar.	
18. In each window logout of dce by issuing the following commands: **kdestroy** **exit**	

4.9.1.4 Test Case 4: Message Passing to Non-Existent Object (TC025.004)

The purpose of this test is to verify that implementation two of the Message Passing Service takes appropriate action when it is unsuccessful in transmitting a message in its allotted time interval. Specifically this test will demonstrate the Message Passing Service's ability to log failed attempt event messages to the MSS management agents, and to store the failed attempt status information for the client to retrieve at its convenience.

Test Configuration:

Hardware: HP and Sun workstations/server HCI, Inter networking subsystem CI (LAN Network resources)-INCI

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions, ECS Common library functions.

Data: N/A

Tools: ret_cli_hp,ret_cli_sun.

Test Input:

A cds entry name must be supplied to the test client application. The client application will also prompt the user for the number of times to try sending the message as well as the time interval between transmission attempts.

Test Output:

The client test driver should issue statements indicating that a message was sent and it is continuing its execution. Subsequently, messages indicating how many transmissions have been attempted and the system time of the current attempt are displayed. When the message object exhausts its allotted transmission attempts an error message will appear in the client window. This message corresponds to the entry that will be logged to the MSS management agent. The uuid of the message object will be displayed along with the return status of the message transmission.

Success Criteria:

The client application sends a message to the server and continues executing. The Message Service will retransmit the message until it exhausts its allotted number of transmission attempts. The Message Service will log the transmission failure to the MSS management agent and the corresponding log message will appear in the client window. The client retrieves the return information indicating that the execution of the message thread was unsuccessful.

Test Case ID: TC025.004	
Test Name: Message Passing to Non-Existent Object	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.
2. Create a session (i.e. a xterm windows) and log into DCE by typing : **dce_login** *username password*	
3. In the window start the client application by typing the following: **ret_cli_hp** *<cds entry name>*	Normally the cds entry name would correspond to the cds entry name under which the server application was registered. Since the server does not exist in this case an entry name should be used that is not registered in the cell's name space.
4. When prompted by the client test driver, enter the number of times to try sending the message and press **Enter**.	
5. When prompted by the client test driver, enter the time interval to wait between transmission attempts and press **Enter**.	
6. Verify the following message appears in the client window indicating the message was issued: "Sending message to *receiver...*"	The message is sent using the EcDcDSyncCom::Send method. (C-CSS-22000)
7. Verify the following message appears in the client window indicating the client has regained control of the process's execution: "Continuing execution..."	This message is displayed immediately after the EcDcDSyncCom::Send method is invoked. (C-CSS-22000) The client test driver will now sleep until the message execution is complete. This is required so the message in step 12 will be as indicated.
8. Verify the following message appears in the client window indicating the message thread is in its initial attempt at execution: "Current transmission attempt: 1." "Time of attempt:*<current time>*"	Messages of this format will continue to appear in the client window until the Message Service has attempted to transmit the message the number of times specified in step 4.(C-CSS-22150)
9. Verify the following message appears in the client window indicating the message thread could not locate the server object: "**DCE Exception: entry not found (dce/rpc)"	This DCE Exception occurs because the server application has not been registered in the cell's namespace. A message of this type will occur every time a transmission attempt fails because the object could not be found.

10. Verify the following message appears in the client window indicating the failed execution of the message thread: " MessageId=EcDcDSyncCom::EvalThread PreviousMessageLink=0 Severity=Error Failed: Invoke - Exhausted the number of tries"	The text of this message corresponds to the text that will be logged to the MSS Management agent. (C-CSS-22065). The EcLgLogger class used by the Message Service actually displays the text to the screen. This requirement is only partial fulfilled because there is no management agent available to verify that the message was actually received by the agent.
11. Verify the following message appears in the client window indicating that the client has obtained an UUID corresponding to the message object: "Message UUID is: *<returned uuid>*"	This UUID corresponds to the message object being transmitted. (C-CSS-22130)
12. Verify the following message appears in the client window indicating the return status of the message corresponds to unsuccessful completion: "Message return status is: -1"	The message/thread return status is stored in the private member "_done" and can be obtained using the message object's Done() member function. A value of 1 indicates the thread completed successfully and -1 indicates the thread failed in some point of its execution. If the thread isn't complete a value of 0 will be returned. (C-CSS-22140)
13. In the window logout of dce by issuing the following commands: **kdestroy** **exit**	

4.9.1.5 Test Case 5: Message Thread Configuration (TC025.005)

The purpose of this test is to verify that implementation two of the Message Passing Service allows the developer to configure certain attributes of the thread used to transmit the message. Specifically, the developer can define the thread scheduling policy, and scheduling priority for each transmission of the message.

Test Configuration:

Hardware: HP and Sun workstations/server HCI, Inter networking subsystem CI (LAN Network resources)-INCI

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions, ECS Common library functions.

Data: N/A

Tools: thr_cli_hp,srv_hp,thr_cli_sun,srv_sun.

Test Input:

A cds entry name must be supplied to the test client application. The client application will also prompt the user for the scheduling priority and scheduling policy of each message sent.

Test Output:

The client test driver should issue statements indicating the thread scheduling priority and policy specified by the user. A statement indicating the message was sent and client execution is continuing will follow. Subsequently it will display the associated uuid for the message object, the return status of the message transmission and ask the user if another thread is to be sent. The server test driver should issue a statement indicating that the message was received and processed for each message sent.

Success Criteria:

For each thread sent, statements are issued indicating the thread policy and priority members for the message object have been appropriately set. The client application sends the messages to the server and continues executing. The client retrieves return information indicating that the execution of each message thread was successful.

Test Case ID: TC025.005	
Test Name: Message Thread Configuration	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.
2. Create two sessions (i.e. open two xterm windows) and log into DCE in both by typing : **dce_login** *username password*	
3. In one window, start the cdsbrowser application in the background by typing: **cdsbrowser &**	Note the cdsbrowser application is available only on HP platforms. When executing this testcase on other platforms create a third session in step 2. In this window log into a HP machine in the same cell as the test machine, set your DISPLAY variable to point back to the test machine, and then execute step 3.
4. Start the test server application by typing the following in one of the test machine windows: **srv_hp** *<cds entry name>*	The cds entry should be placed in a directory in which the executor of the test has write permission. For example, in the relacell the following is a valid entry name: */.:/RelA/msg_server* where */.:/RelA/* is the directory and *msg_server* is the server name. A message stating "The server is now listening...." should appear in the server window indicating that the server is running.
5. Verify the server object has been registered in the CDS by clicking the **tree->update** button in the cdsbrowser's toolbar.	The server object should now be displayed inside the user-specified directory (e.g. underneath the */.:/RelA/* directory)
6. In the second window start the client application by typing the following: **thr_cli_hp** *<cds entry name>*	The cds entry name should correspond to the cds entry name specified in step 4.
7. When prompted by the client test driver, enter the scheduling policy attribute for the message thread and press **Enter**.	The four choices are **SCHED_FIFO, SCHED_RR,SCHED_BG_NP,** and **SCHED_OTHER (SCHED_FG_NP)** The default is **SCHED_OTHER**.

8. When prompted by the client test driver, enter the scheduling priority attribute for the message thread and press **Enter**.	The OODCE thread priority types are **Pthread_pri_min, Pthread_pri_low, Pthread_pri_mid, Pthread_pri_hi,** and **Pthread_pri_max.**
9. Verify the following message appears in the client window indicating the thread has the appropriate scheduling policy and priority attributes: "The thread's scheduling policy is: *<appropriate policy>*" "The thread's scheduling priority is: *<appropriate priority value>."*	The values displayed are obtained via the message object's GetPolicy() and GetPriority() functions. The private members returned by these members are used as arguments to the DCEPthread::Priority(), and DCEPthread::Scheduling() functions. It is assumed that the DCE software appropriately schedules the execution of the threads when passed the proper arguments. Therefore, the successful storage of the user specified attribute is an acceptable success criteria for this testcase.
10. Verify the following message appears in the client window indicating the message was issued: "Sending message *<message #>* to *receiver...*"	The message is sent using the EcDcDSyncCom::Send method. (C-CSS-22000)
11. Verify the following message appears in the client window indicating the client has regained control of the process's execution: "Continuing execution..."	This message is displayed immediately after the EcDcDSyncCom::Send method is invoked. It should appear prior to the message displayed by the server in the next step. (C-CSS-22000) The client test driver will now sleep until the message execution is complete. This is required so the message in step 14 will be as indicated.
12. Verify the following message appears in the server window indicating the server has received the message: "Message *<message #>* received from client..."	
13. Verify the following message appears in the client window indicating that the client has obtained an UUID corresponding to the message thread: "Message *<message #>* UUID is: *<returned uuid>*"	This UUID corresponds to the object UUID of the message thread. (C-CSS-22130)
14. Verify the following message appears in the client window indicating the return status of the message corresponds to successful completion: "Message *<message #>* return status is: 1"	The message/thread return status is stored in the private member "_done" and can be obtained using the message object's Done() member function. A value of 1 indicates the thread completed successfully and -1 indicates the thread failed in some point of its execution. If the thread isn't complete a value of 0 will be returned. (C-CSS-22140)
15. Repeat steps 7-14 for each scheduling attribute.	Following step 14, the test driver will ask the user if they wish to send another thread. A thread should be sent for each combination of scheduling policy and priority.
16. Kill the server process by issuing a kill signal in the server window. (e.g. **Ctrl-C)**	
17. Remove the server object from the cds namespace by highlighting the server object in the cdsbrowser and clicking on the **options->delete** entry button in the toolbar.	

18. In each window logout of dce by issuing the following commands: **kdestroy** **exit**	

4.9.1.6 Test Case 6: Transmission of Multiple Messages (TC025.006)

The purpose of this test is to verify that implementation two of the Message Passing Service is capable of transmitting multiple messages within an application. Specifically this test will show that an application may transmit messages concurrently by utilizing multiple instances of the message object.

Test Configuration:

Hardware: HP and Sun workstations/server HCI, Inter networking subsystem CI (LAN Network resources)-INCI

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions, ECS Common library functions.

Data: N/A

Tools: multi_cli_hp, multi_srv_hp, multi_cli_sun, multi_srv_sun.

Test Input:

A cds entry name must be supplied to the test client application. The user will be asked for the number of messages to send if it is not specified on the command line.

Test Output:

The client test driver should issue a statement indicating that the specified number of messages were sent and it is continuing its execution. The server test driver should issue statements indicating the time each message was received and processed. For each message the client will then display the transmission number, the uuid, the calculation performed, the time the message was received and processed by the server, and its return status.

Success Criteria:

The client application sends a message to the server and continues executing. The client receives return information from each message object indicating its transmission was successful.

Test Case ID: TC025.006	
Test Name: Transmission of Multiple Messages	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.
2. Create two sessions (i.e. open two xterm windows) and log into DCE in both by typing : **dce_login** *username password*	
3. In one window, start the cdsbrowser application in the background by typing: **cdsbrowser &**	Note the cdsbrowser application is available only on HP platforms. When executing this testcase on other platforms create a third session in step 2. In this window log into a HP machine in the same cell as the test machine, set your DISPLAY variable to point back to the test machine, and then execute step 3.
4. Start the test server application by typing the following in one of the test machine windows: **multi_srv_hp** *<cds entry name>*	The cds entry should be placed in a directory in which the executor of the test has write permission. For example, in the relacell the following is a valid entry name: */.:/RelA/msg_server* where */.:/RelA/* is the directory and *msg_server* is the server name. A message stating "The server is now listening...." should appear in the server window indicating that the server is running.
5. Verify the server object has been registered in the CDS by clicking the **tree->update** button in the cdsbrowser's toolbar.	The server object should now be displayed inside the user-specified directory (e.g. underneath the */.:/RelA/* directory)
6. In the second window start the client application by typing the following: **multi_cli_hp** *<cds entry name>*	The cds entry name should correspond to the cds entry name specified in step 4.
7. When prompted by the client test driver, enter the number of messages to transmit and press **Enter**.	
8. Verify the following message appears in the client window indicating the messages were issued: "*<#>* messages sent... continuing execution...."	The # sign should correspond to the number entered in step 7. The messages are sent using each object's EcDcDSyncCom::Send method. (C-CSS-22010) Each message object calls this function within a separate thread created by the test client application. The messages consist of two randomly generated numbers that are sent to the server where they are multiplied and their product is returned to the message object.
9. Verify a message similar to the following appears in the server window for each message sent: "Transmission *<transmission #>* received from client at *<time string>*"	The numbering of these messages may appear out of order. This is due to the fact that the transmission number corresponds to the order in which each message object's PreInvoke() method was first entered and not when the rpc call was actually made. The rpc call occurs within the Invoke() method.

10. Verify a message similar to the following appears in the server window for each message sent: "Transmission *<transmission #>* concluded processing at *<time string>*"	NOTE: Each rpc call sleeps during its execution for a distinct amount of time. The sleeping period is based upon its transmission number i.e. (transmission 1 will sleep for 1 second, transmission 2 will sleep for 2 seconds). This should, but not necessarily, cause the rpc's to finish processing in sequence.
11. Verify the following message appears in the client window for each message sent: "Message *<message #>* I am transmission: *<attempt #>* My uuid is: < *<returned uuid>>* I calculated: *<calculation string>* I was received on *<time string>* I was returned on *<time string>* Message's return status:1"	The message number corresponds to the order in which the message objects were created in the test driver. The transmission number corresponds to the order in which each message thread called the PreInvoke() function. Therefore these two numbers may not be the same. The indented portion of the message corresponds to the results retrieved from each message object via its GetResults() method. (C-CSS-22010). The UUID corresponds to the message object being transmitted. (C-CSS-22130). Normally the received time string will be identical for all the messages. However, if a large number (e.g. 50) of rpcs are sent to the same server at the same time then the server request queue may fill up and additional connections will be refused. A DCE object not found message should appear in this case for each failed rpc. In this driver, each message object will retry sending the rpc 10 times before considering that message transmission a failure. The message/thread return status is stored in the private member "_done" and can be obtained using the message object's Done() member function. A value of 1 indicates the thread completed successfully and -1 indicates the thread failed in some point of its execution. If the thread isn't complete a value of 0 will be returned. (C-CSS-22140)
12. Remove the server object from the cds namespace by highlighting the server object in the cdsbrowser and clicking on the **options->delete** entry button in the toolbar.	
13. In each window logout of dce by issuing the following commands: **kdestroy** **exit**	

4.9.2 Communications Infrastructure 2 Build (BC008)

The Communications Infrastructure 2 build is the integration of the Communication Infrastructure 1 build and the Message Passing 2 thread. The Communication Infrastructure 1 build is an aggregation of the Message Passing 1 thread, the Security Authentication thread, and the Directory naming thread. This build demonstrates the capability of integrating this functionality into a sub component of the overall subsystem.

4.9.2.1 Test Case 1: Communication Infrastructure 1 Regression Test (BC008.001)

This test is a regression test of the overall functionality of the Communication Infrastructure 1 build. This functionality includes the ability to support basic network security authentication, deferred synchronous message passing, and access to the cell directory namespace.

The objective of this test is to verify that all of these functionality remains available and unchanged upon integration of the Communication Infrastructure 2 Build.

4.9.2.2 Test Case 2: Guaranteed Asynchronous Message Passing (BC008.002)

The purpose of this test is to verify that implementation one of the Message Passing Service has the capability to transmit messages/data via the asynchronous message passage mode of communication and guarantee the successful completion of the transmission. Specifically, this test demonstrates that implementation two of the Message Passing Service:

- provides a non blocking API for clients(senders) to send messages to servers(receivers) asynchronously.
- Supports guaranteed delivery of the message to the receiver.
- can associate the receiver and the message object to a return value that is available upon request to the sender.
- provides an API for senders to obtain return information concerning the transmission of the message.

Test Configuration:

Hardware: HP and Sun workstations/servers HCI, Inter networking subsystem CI (LAN Network resources), INCI

Software: DCE/OODCE, HP cdsbrowser, DNS, CSS APIs & library functions

Data: Tester supplied input

Tools: guar_cli_hp, srv_hp, guar_cli_sun, srv_sun.

Test Input:

The name under which the test server application (e.g. srv_sun) will be registered in the cell directory service must be supplied to both the test server and client applications.

Test Output:

The client test driver should issue statements indicating that it is initializing the message service, creating a callback receiver and its associated queue, along with a callback object. It will then

create a sending session list, add the callback receiver to the session list, send the message and it continue its execution. It should then display the uuid of the message object, the uuid of the receiver, an acknowledgment that the message was delivered to the intended receiver, and the return status of the message transmission. The server test driver should issue a statement indicating that the message was transferred.

Success Criteria:

The client application sends a message to the server and continues executing and subsequently receives an acknowledgment indicating that the message was transmitted successfully and the message was retrieved from the queue.

Test Procedures:

Build Test Case ID: BC008.002	
Test Name: Guaranteed Asynchronous Messaging Passing	
Test Steps:	Comments:
1. Log into an HP-UX machine (e.g., csse3hp)	The chosen machine should be a properly configured DCE client host. When executing this testcase on another platform (e.g. Sun) substitute the appropriate platform everywhere HP specific hardware and software is specified unless explicitly instructed to do otherwise.
2. Create two sessions (i.e. open two xterm windows) and log into DCE in both by typing : **dce_login** *username password*	
3. In one window, start the cdsbrowser application in the background by typing: **cdsbrowser &**	Note the cdsbrowser application is available only on HP platforms. When executing this testcase on other platforms create a third session in step 2. In this window log into a HP machine in the same cell as the test machine, set your DISPLAY variable to point back to the test machine, and then execute step 3.
4. Start the test server application by typing the following in one of the test machine windows: **srv_hp** *<cds entry name>*	The cds entry should be placed in a directory in which the executor of the test has write permission. For example, in the relacell the following is a valid entry name: */.:/RelA/msg_server* where */.:/RelA/* is the directory and *msg_server* is the server name. A message stating "The server is now listening...." should appear in the server window indicating that the server is running.
5. Verify the server object has been registered in the CDS by clicking the **tree->update** button in the cdsbrowser's toolbar.	The server object should now be displayed inside the user-specified directory (e.g. underneath the */.:/RelA/* directory)

6. In the second window start the client application by typing the following: **cli_hp** *<cds entry name>*	The cds entry name should correspond to the cds entry name specified in step 4.
7. Verify a message indicating that the Message Passing service has been initialized appears in the client window.	
8. Verify a message indicating that a callback object has been created appears in the client window.	
9. Verify a message indicating that a callback receiver has been created appears in the client window.	
10. Verify a message indicating that a session list has been created appears in the client window.	
11. Verify a message indicating that the callback receiver has been added to the session list appears in the client window.	
12. Verify a message indicating that the message has been sent appears in the client window.	
13. Verify a message indicating the client has regained control of the process's execution appears in the client window.	
14. Verify a message indicating the transfer server has received the message appears in the server window.	
15. Verify a message appears in the client window indicating that the client has obtained an UUID corresponding to the message object.	
16. Verify a message appears in the client window indicating that the client has obtained an UUID corresponding to the receiver.	
17. Verify a message appears in the client window indicating that the client has received acknowledgment that the message was delivered successfully to the intended receiver.	
18. Verify a message appears in the client window indicating the return status of the message transmission.	
19. Verify a message appears in the client window indicating the message was retrieved from the queue.	
20. Kill the server process by issuing a kill signal in the server window. (e.g. **Ctrl-C)**	
21. Remove the server object from the cds namespace by highlighting the server object in the cdsbrowser and clicking on the **options->delete** entry button in the toolbar.	
22. In each window logout of dce by issuing the following commands: **kdestroy** **exit**	

4.10 Management Services 2 Tests

Management Services 2 tests consist of the following builds and threads:

- Software Fault Management Thread (TC027)
- Security Management Thread (TC028)
- Operator Role Management Thread (TC029)
- Management Data Access II Thread (TC030)
- Baseline Management Thread (TC031)
- Management Services 2 Build (BC009)

4.10.1 Software Fault Management Thread (TC027)

The Fault Management Application Service (FMAS) provides the capability to detect, diagnose, isolate, and recover from faults that occur in the managed objects within ECS. The entities or managed objects in ECS that need to be monitored for faults include network devices (such as hosts, hubs and routers), systems software (databases and middleware such as DCE) and applications (such as the Planning Subsystem and the Data Server Subsystem). Fault management encompasses activities such as the ability to trace faults through the system, to execute diagnostic tests, and to initiate corrective or recovery actions upon the isolation of errors in order to correct faults. This thread will primarily test aspects of software fault management.

4.10.1.1 Test Case 1: Software Fault Categories (TC027.001)

This test case verifies the ability of the FMAS software to maintain and manage various areas of system faults for M&O staff, which includes assigning severity levels and categories to faults, enabling or disabling the logging of specific faults or fault types, detect software faults, logging faults and to maintain and update system fault policies and procedures to include fault identification, fault priorities, and corrective actions.

Test Configuration:

Hardware: Workstations, printers, etc.

Software: HpOpenView Network Node Manager

Data: None

Tools: A script(s) to generate SW faults within the ECS.

Test Input:

Any user-initiated faults.

Test Output:

Message logs that show various faults, their severity level, and their categories.

Success Criteria:

This test is successful when the message log has been verified to contain the faults, and the respective severity levels and categories to which they belong.

Test Procedures:

Test Case ID: TC027.001	
Test Name: Software Fault Categories	
Test Steps:	Comments
1. Login to FMAS workstation to Categorize faults received from the DAACs and the SMC	C-MSS-60030, C-MSS-60020,
2. Assign severity levels and categories to the faults.	C-MSS-60040
3. Enable or disable faults for audit logging.	C-MSS-60060
4. Verify software faults detected and properly logged.	C-MSS-60190
5. Verify disabled faults not logged.	C-MSS-60050
6. Maintain and update system fault policies and procedures to include fault identification, fault priorities, and corrective actions.	C-MSS-60070
7. Verify updates reflected in menu prompts and audit logs.	

4.10.1.2 Test Case 2: Software Fault Metrics (TC027.002)

This test case verifies that the FMAS can demonstrate the capability to manipulate thresholds on performance metrics, poll for detection of fault/performance information via SNMP queries, set the polling frequency for fault and performance data. FMAS shall also generate fault notification(s) when performance thresholds have been exceeded and generate fault metric reports. FMAS shall detect communication software version mismatch, software configuration and communications protocol errors. Database faults shall be detected along with missing or looping software errors. FMAS shall generate performance/error data reports interactively and on a scheduled basis, build history reports of different types of errors and distribute the reports by redirecting reports to a console, disk file, or printer.

Test Configuration:

Hardware: Workstations, printers, etc.

Software: HpOpenView Network Node Manager

Data: None

Tools: A script to generate SW faults within the ECS.

Test Input:

Any user-initiated faults.

Test Output:

Logs that show the various fault metric information.

Success Criteria:

This test is successful when a report is generated that contains software fault metric information properly reflecting the simulated faults.

Test Procedures:

Test Case ID: TC027.002	
Test Name: Software Fault Metrics	
Test Steps:	Comments
1. Manipulate thresholds on performance metrics.	C-MSS-60080
2. Poll for detection of fault/performance information.	C-MSS-60100
3. Set polling frequency of fault/performance information.	C-MSS-60120
4. Generate a fault notification when performance threshold has been exceeded.	C-MSS-60140
5. Generate fault metric reports.	
6. Detect communication software version mismatch errors.	
7. Detect communication software configuration errors.	
8. Detect errors in the communications protocol.	
9. Detect database faults.	
10. Detect errors with missing or looping software.	C-MSS-60130 partial
11. Generate performance/error data reports interactively and on a scheduled basis build history reports of different types of errors.	C-MSS-60610, C-MSS-60600
12. Redirect reports to a console, disk file, or printer.	C-MSS-60620

4.10.1.3 Test Case 3: Multiple Sources of Software Faults (TC027.003)

This test case verifies that FMAS can demonstrate the capability of overall network management and monitoring of system performance from both the local site(s) and SMC levels. This test will demonstrate the capability to receive from and distribute the necessary notification to designated M&O staff or administrative personnel including summary reports to the SMC.

Test Configuration:

Hardware: Workstations, printers, etc.

Software: HpOpenView Network Node Manager

Data: NONE

Tools: A script(s) to generate SW faults within the ECS.

Test Input:

Any user-initiated faults.

Test Output:

Output includes notification that a software fault has occurred.

Success Criteria:

This test is successful when the FMAS is able to monitor fault information from multiple sites. This information can be from either SNMP agents or applications.

Test Procedures:

Test Case ID: TC027.003	
Test Name: Multiple Sources of Software Faults	
Test Steps:	Comments
1. Receive fault notification.	
2. Receive fault notification from SNMP agents and other fault management applications.	C-MSS-60160, C-MSS-60150
3. Monitor fault performance information.	
4. Receive the above information from multiple sites.	C-MSS-60110
5. Request fault notification information from multiple sites and other external systems.	C-MSS-60170
6. Receive the above information in summarized form from multiple sites.	C-MSS-60180
7. Retrieve records of detected faults.	C-MSS-60400

4.10.1.4 Test Case 4: Software Fault Attributes (TC027.004)

This test case verifies the ability of the FMAS to generate customized types of fault notifications (color, pop-up, logging to a disk file, and audible alert) to be enabled for various faults. Maintain a distribution list of registered users to be notified in the event of a fault and notify those users in the event of a fault. FMAS shall also generate notifications within specified time constraints set by the network monitoring policies.

Test Configuration:

Hardware: Workstations, printers, etc.

Software: HpOpenView Network Node Manager

Data: NONE

Tools: A script(s) to generate SW faults within the ECS.

Test Input:

Any user-initiated faults.

Test Output:

Output includes notification that a software fault has occurred.

Success Criteria:

This test is successful when the FMAS notifies operators of a fault within the minimum allowed time. This notification will be via audible and color alerts, and pop-up windows. Also, the fault information is logged to a disk file.

Test Procedures:

Test Case ID: TC027.004	
Test Name: Software Fault Attributes	
Test Steps:	Comments
1. Create a distribution list of registered users to be notified during specific fault event types, including M&O and any external service providers.	C-MSS-60210
2. Generate the different types of fault events to generate notifications within a specified time constraint.	Fault types include color, pop-up, logging to a disk file, and audible alert. C-MSS-60200
3. Verify that the fault notifications are properly distributed to the registered users in the list.	C-MSS-60220
4. Verify faults are logged in the event log.	
5. Verify fault notification time-stamps for notification times received within five minutes of detection.	C-MSS-60230
6. Verify logs and reports generated are accurate.	C-MSS-60395

4.10.2 Security Management Thread (TC028)

The overall objective of Security Management Thread is to verify that the system (i.e. SMAS component) provides for the management of the ECS security mechanisms used to protect and control access to ECS resources. Specifically, both technical and administrative types of security mechanisms, such as user authentication, the maintenance of authorization facilities, the maintenance of security logs, and intrusion detection and recovery procedures for the following ECS items will be verified:

- Adequacy of security provided for the network
- Adequacy of security provided for the distributed communications
- Adequacy of security provided for the host

The SMAS provides security management at the ECS sites and the SMC.

4.10.2.1 Test Case 1: ECS Security Access (TC028.001)

This test case verifies the ability of the SMAS to:

- specify privileges for authorized users
- allow/deny access to the ECS host based on privileges

Test Configuration:

Hardware: ECS Management workstation

Software: HAL DCE Cell Manager, ACL_EDIT, OS ACL, MSS Management Software CI-MCI

Data: Authorized ECS user account

Tools: CRACK, Npasswd, SATAN, TCP Wrappers

Test Input:

Invalid user accesses to the ECS

Test Output:

Logs that contain user group names and privileges.

Success Criteria:

This test is successful when privileges for users can be specified and user can be denied (unauthorized) or allowed access (authorized) to the ECS

Test Procedures:

Test Case ID: TC028.001	
Test Name: ECS Access	
Test Steps:	Comments
1. Create an ECS user account, assign a group and login directory.	C-MSS-70010, C-MSS-70020,
2. Attempt login to ECS client with invalid userid.	
3. Verify userid is denied and notification is displayed to the user.	
4. Verify login attempt is logged in the MSS audit security log.	
5. Attempt login to ECS client with valid userid and invalid password.	Use SATAN, CRACK, Npasswd
6. Verify userid is denied and notification is displayed to the user.	
7. Verify login attempt is logged in the MSS audit security log.	C-MSS-70450
8. Login into ECS client with a valid userid and valid password.	Authorized ECS user login.
9. Verify user is granted access to ECS account.	TCP Wrapper C-MSS-70100, C-MSS-70110, C-MSS-70120

4.10.2.2 Test Case 2: Site Security Functions (TC028.002)

This test case verifies the ability of the SMAS to:

- simulate a scheduled security tests at the site
- send site test results to the EMC SMAS (SMC)
- receive security notification events from the sites

Test Configuration:

Hardware: MSS CI (local management server, management workstation)- MHCI, printers, etc., Inter networking subsystem CI (LAN resources)-INCISMC

Software: HAL DCE Cell Manager, DCE ACL_EDIT, OS ACL, Kerberos network utilities, MSS Management Software CI-MCI

Data: Valid/Invalid user accounts

Tools: CRACK, Npasswd, SATAN, TCP Wrappers

Test Input:

Simulated security violation test scenario.

The security protection mechanisms tested shall include:

- password audit (Compliance Test)
- file system integrity check(Compliance Test)
- user privileges audit(Compliance Test)
- resource access control information(Compliance Test)
- security notification events
- site test result file

Test Output:

All security violation attempts simulated are accurately detected, notification events received, security audit logs are updated and the results get forwarded to the SMC.

Success Criteria:

This test is successful when site security tests can be performed, security notification events are received, and the results can be forwarded to the EMC (SMC).

Test Procedures:

Test Case ID: TC028.002	
Test Name: Site Security Access	
Test Steps:	Comments
1. Login to ECS site.	
2. Simulate a site security violation test.	Record all security configuration settings.
3. Verify security control mechanisms accurately detect security violation attempts into the site network.	Use SATAN, Npasswd, CRACK C-MSS-70300

4. Verify proper violation notifications are received at the site and SMC.	Use TCP Wrappers C-MSS-70430, C-MSS-70440
5. Verify messages are sent to the appropriate security audit logs.	

4.10.2.3 Test Case 3: Site Security Intrusion Functions (TC028.003)

This test case verifies the ability of the SMAS to:

- analyze security audit trails to detect intrusions and their nature
- designate users and M&O staff to receive notification upon an intrusion
- generate a notification within five minutes of detection
- log incoming access attempts

Test Configuration:

Hardware: MSS CI (local management server, management workstation), MHCI, Inter networking subsystem CI (LAN resources), INCI

Software: HAL DCE Cell Manager, ACL_EDIT, OS ACL, MSS Management Software CI-MCI

Data: Valid/Invalid site accounts

Tools: TCP Wrapper, Tripwire

Test Input:

Security intrusion (e.g. Break-ins Intrusion Detection Test)

Test Output:

Security mechanism accurately detects intrusion violations due to improper access attempts to files, login failures and break-ins by generating notifications to the appropriate destinations.

Success Criteria:

This test is successful when the following occurs:

- security intrusions such as login failures, unauthorized access to ECS resources, and break-ins are detected, logged, analyzed, and notified by the system.
- the system generates notification within a maximum of five minutes following the detection of an intrusion.

Test Procedures:

Test Case ID: TC028.003	
Test Name: Site Security Intrusion Functions	
Test Steps:	Comments
1. Login to EMC site.	
2. Simulate a site intrusion violation.	Use Tripwire
3. Designate users and M&O staff to receive notification upon an intrusion.	C-MSS-70430, C-MSS-70440,
4. Verify security control mechanisms accurately detect security intrusion attempts into the site network.	
5. Verify proper violation notifications are received at the site and SMC within five minutes of detection.	C-MSS-70320, C-MSS-70460
6. Verify messages are logged to the appropriate security audit logs.	
7. Analyze the nature of the intrusion via Tripwire, TCP wrappers and security audit data.	
8. Verify security data is accurate.	C-MSS-70310

4.10.2.4 Test Case 4: Security Recovery Procedures (TC028.004)

This test case verifies the ability of the SMAS to:

- coordinate the recovery of EMC security compromises
- coordinate the recovery of site compromises
- provide office automation tools to enable generation of directives for recovery from security events

Test Configuration:

Hardware: MSS CI (local management server, management workstation), MHCI, Inter networking subsystem CI (LAN resources), INCI

Software: HAL DCE Cell Manager, ACL_EDIT, OS ACL, MSS Management Software CI-MCI;

Data: None

Tools: Office Automation tools for compromise recovery, CRACK, Npasswd, SATAN, TCP Wrappers, Tripwire

Test Input:

Operator actions are performed to create a security alert and site compromise. Recovery procedures are then followed and analyzed for effectiveness.

Test Output:

Output includes alarms generated as a result of security alert and site compromise.

Success Criteria:

This test is successful when security recovery instructions and directives are:

- generated with the use of office automation tools
- performed successfully so that the security compromising elements or areas are isolated and/or eliminated from the system.

Test Procedures:

Test Case ID: TC028.004	
Test Name: Security Recovery Procedures	
Test Steps:	Comments
1. Login to ECS site.	
2. Create a security compromise by exceeding a configured security control mechanism.	Alert controls are managed by site policies for security.
3. Verify Office Automation tools provide for the generation of directives for recovery from the security event.	Review documented recovery procedures provided by office automation tools.
4. Verify security compromising elements or areas are isolated and/or eliminated from the system.	C-MSS-70510

4.10.2.5 Test Case 5: Security Report Generation (TC028.005)

This test case verifies the ability of the SMAS to:

- generate reports on all security violations
- redirect report output to a console, printer, or disk file

Test Configuration:

Hardware: MSS CI (local management server, management workstation), MHCI, Inter networking subsystem CI (LAN resources), INCI

Software: HAL DCE Cell Manager, ACL_EDIT, OS ACL, MSS Management Software CI-MCI;

Data: None

Tools: Office Automation tools, CRACK, Npasswd, SATAN, TCP Wrappers, Tripwire

Test Input:

Security violation such as incorrect/incomplete login, unauthorized access to ECS resources, and break-ins.

Test Output:

Report for the security violation commensurable with the inputs.

Success Criteria:

This test is successful when correct security violation reports are generated and redirected on either a workstation console, disk file, or printer.

Test Procedures:

Test Case ID: TC028.005	
Test Name: Security Report Generation	
Test Steps:	Comments
1. Generate a security violation report.	
2. Verify security report generation policies and procedures are being implemented.	
3. Verify security violation reports are generated and redirected to their proper destination on either a workstation console, disk file, or printer.	C-MSS-70130, C-MSS-70700, C-MSS-70710, C-MSS-70720

4.10.2.6 Test Case 6: Security Management Interface (TC028.006)

This test case verifies the user's ability to:

- access the SMAS from a command line interface
- access the SMAS from a GUI
- manage the site's authentication, authorization, and network security databases

Test Configuration:

Hardware: MSS CI (local management server, management workstation), MHCI, Inter networking subsystem CI (LAN resources), INCI

Software: HAL DCE Cell Manager, DCE ACL_EDIT, OS ACL, Router_ACL, TCP Wrapper, Tripwire, GUI S/W, MSS Management Software CI-MCI, Authentication, Authorization and Network Security Database

Data: None

Tools: None

Test Input:

Test inputs include command line keyins for access to Authentication, Authorization, or security database information via SMAS from an Xterm or GUI workstation.

Test Output:

Outputs include indication of access to authentication, authorization and network database information in the SMAS.

Success Criteria:

This test is successful when the database can be accessed and the required functions are performed by the interfaces listed above.

Test Procedures:

Test Case ID: TC028.006	
Test Name: Security Management Interface	
Test Steps:	Comments
1. Verify that a security user (M&O) has access via security management interfaces to perform all security administration, configuration, and maintenance tasks.	
2. Verify command line interfaces allow access to security management database.	
3. Verify GUI interfaces allow access to security management database.	

4.10.3 Operator Role Management Thread (TC029)

The purpose of the Operator Role Management thread is to verify the basic capabilities provided to the operator. These capabilities include the ability to create a new operator account upon request and assign appropriate role(s) by the system administrator at each site. All available role(s) comes with certain privileges and each operator will have a Dce_Login account to access the Desktop window for Operator Role Management to perform the necessary role(s) functions for their task.

4.10.3.1 Test Case 1: Retrieving User Activity Data (TC029.001)

This test demonstrates that the system administrator at each site has the capability to create a new operator account request from the user service group and provide a operator profile with the appropriate role(s) information.

Test Configuration:

Hardware: Local System Management (ECS) MSS Server (HP) and Workstation

Software: Tivoli COTS and Developed DCE

Data: Input from GUI screen, Sybase Database, Flat file

Tools: Sybase

Test Input:

Operator account information from request.

Test Output:

Verification of inputs from Operator Role Management database regarding new operator request and assign role(s) information.

Success Criteria:

This test is considered successful when the Operator Role Management database returns the created new operator request information.

Test Procedures:

Test Case ID: TC029.001	
Test Name: Create A New Oper. Acct. Request	
Test Steps:	Comments:
1. Receive a request for a New Operator Account.	The request will include: User Name, Office No., Work Phone, Home Phone, Role(s) to be assign, DCE Group/DCE Organization.
2. Login as a Systems Personnel capable of creating a new operator account.	
3. Start the Tivoli server.	
4. Verify server is running.	
5. Bring up "Add Record To Profile" Screen.	Add Record To Profile window appears.
6. Under user profile "Add Record to Profile".	
7. Enter the ***<User Name>***.	
8. Select [**Use Defaults**] at bottom of screen.	
9. Enter remaining user information.	The selections are: Login Name, User id, Primary Group, Office No., Work No., Home No., and Selected Role(s).
10. Select [**Add & Close**] at bottom of screen.	
11. From the Profile pull down menu on User Profile Properties window.	User Profile Properties Window appears.
12. Select [**Save**].	Saves the user profile information.
13. Select [**Distribute**].	
14. From the Distribute window, Select [**Distribute & Close**]	
15. Enter the required role(s).	If a Role(s) requires Tivoli.
16. Select the Administrator icon and add user login to the appropriate available role(s).	
17. Move to appropriate Role(s) icon and using <RMB>, then Select "Edit Logins".	RMB= Right Mouse Button
18. Exit out Add Record To User Profile screen.	

4.10.3.2 Test Case 2: Verification of New Operator Account (TC029.002)

This test verifies that a new operator account request is entered in the system with all the appropriate operator information and role(s) provided for accuracy.

Test Configuration:

Hardware: Local System Management (ECS) MSS Server (HP) and Workstation

Software: Tivoli COTS and Developed DCE

Data: Input from GUI screen, Sybase Database, Flat file

Tools: Sybase

Test Input:
Operator profile account request information.

Test Output:

Verification of the operator profile account request information has all the appropriate data for that assign account dump to the screen.

Success Criteria:

This test is considered successful when the system administrator can verify that the new operator request login is correct and the assigned role(s) created can be accessed with the proper functions.

Test Procedures:

Test Case ID: TC029.002	
Test Name: Verification of New Oper. Account	
Test Steps:	Comments:
1. Login a Unix Account.	
2. Verify that you can access the new operator account through the Unix login.	
3. Verify that the ECS login window appear.	
4. Verify the role(s) that were set while creating the account show up.	Bring up each Role(s) individually for correctness.
5. Verify access to the new operator DCE (ECS) login account.	
6. Verify ECS Desktop window pops up.	ECS Desktop screen appears.
7. Verify that he proper role(s) privileges as define associated with the Desktop is correct.	All the role(s) appear on the Desktop. Bring up each individually.
8. Exit out of all new operator accounts.	ECS Desktop, DCE_login and Unix Account.

4.10.3.3 Test Case 3: Access Operator Role(s) With Privileges (TC029.003)

This test verifies that operators with privileges can access restricted role(s) an able to execute their operations. This test will also verify that unauthorized operators are denied access to restricted operator role(s).

Test Configuration:

Hardware: Local System Management (ECS) MSS Server (HP) and Workstation

Software: Tivoli COTS and Developed DCE

Data: Input from GUI screen, Sybase Database, Flat file

Tools: Sybase

Test Input:
Operator role(s) information with defined privileges.

Test Output:

Outputs to this test case include a screen dump of the operator role(s) with defined privileges.

Success Criteria:

This test is successful when an operator can only access and execute through their assigned role(s).

Test Procedures:

Test Case ID: TC029.003	
Test Name: Access Oper. Role With Privileges	
Test Steps:	Comments:
1. Login as a Systems Personnel capable of accessing an account with privileges.	
2. Verify that you can access the operator account through the Unix login.	
3. Verify that the ECS login window appear.	
4. Verify the role(s) that were set you cannot access beyond there assign privileges with additional privileges.	Bring up each Role(s) individually.
5. Verify access to the operator DCE (ECS) login account.	
6. Verify ECS Desktop window pops up.	ECS Desktop screen appears.
7. Verify that the proper role(s) privileges as define associated with the Desktop cannot be access beyond there assign privileges with additional privileges.	All the role(s) appear on the Desktop. Bring up each individually.
8. Exit out of all new operator accounts.	ECS Desktop, DCE_login and Unix Account.

4.10.3.4 Test Case 4: Access Operator Role(s) Without Privileges (TC029.004)

This test demonstrates the capability for an operator to access only their operator role(s) provided by their defaults.

Test Configuration:

Hardware: Local System Management (ECS) MSS Server (HP) and Workstation

Software: Tivoli COTS and Developed DCE

Data: Input from GUI screen, Sybase Database, Flat file

Tools: Sybase

Test Input:

Operator role(s) information.

Test Output:

Outputs to this test case include a screen dump of the operator role(s) with basic functionality.

Success Criteria:

This test is successful when all of the desired operator role(s) with basic functionality cannot go beyond their role(s).

Test Procedures:

Test Case ID: TC029.004	
Test Name: Access Oper. Role W/O Privileges	
Test Steps:	Comments:
1. Login as a Systems Personnel capable of accessing an account without privileges.	
2. Verify that you can access the operator account through the Unix login.	
3. Verify that the ECS login window appear.	
4. Verify the basic role(s) that were set up you cannot successfully go beyond there functionality.	Bring up each Role(s) individually.
5. Verify access to the operator DCE (ECS) login account.	
6. Verify ECS Desktop window pops up.	ECS Desktop screen appears.
7. Verify that the proper basic role(s) as define associated with the Desktop cannot be access beyond there assign functionality.	All the role(s) appear on the Desktop. Bring up each individually.
8. Exit out of all new operator accounts.	ECS Desktop, DCE_login and Unix Account.

4.10.4 Management Data Access 2 Thread (TC030)

The Management Data Access 2 Thread is to verify the full functionality of the MDA service available for users and applications via APIs.

This thread verifies the ability of the MDA service to:

- allow the viewing and updating of all of its schedule configuration parameters (i.e. time intervals, absolute time specifications, size threshold parameters…) through the MDA's graphical user interface
- transfer and load the individual management data log files from each managed host, to the MSS server at each DAAC
- transfer management data at the DAACs site to the SMC, on a scheduled basis
- retrieve the requested data and display it
- process the management data log files for the purposes of longer term analysis and reporting
- update or alter tables and fields in a database
- allow access to the management data while still maintaining the integrity of the management database.

4.10.4.1 Test Case 1: MDA GUI for Schedule & Config. (TC030.001)

This test case verifies the capability of the MDA Graphical User Interface (GUI) to allow a user to display and change the MDA configuration. The user can insert new configuration data for the configuration entry, the processing time, and the archive time. The user can also remove entry from the configuration list and update the configuration data.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MCI: MsMdProcessLog, MsMdEventField, MsMdMetrics, MsMdLoadMdb

Data: XRunner script, User input data

Tools: XRunner 4.0, MDA user interface GUI

Test Input:

Inputs to this test case include keyins to open the MDA user interface GUI, inserting configuration data, processing time, archive time.

Test Output:

Outputs to this test case include new configuration data.

Success Criteria:

This test is successful when the MDA user interface displays added new data, removed data, change.

Test Procedures:

Test Case ID: TC030.001	
Test Name: User Interface for Schedule & Configuration	
Test Steps:	Comments:
1. Log onto the MSS SUN5 Server	This test can also be done on the MSS HP Server. Note: At the present time, there isn't a n HP XRunner license.
2. Set the Display environment: **setenv DISPLAY <*host ip address*>:0.0**	The environment Display is to ensure that all responses be returned to the machine screen that the user logs onto.
3. Change to directory structure where the software resides: **cd (for exaple: *cd /usr/testa*)**	
4. Bring up another MSS SUN5 Server xterm	
5. Set the Display environment: **setenv DISPLAY <*host ip address*>:0.0**	

6. Start the XRunner application: **xrun &**	The XRunner application will run in the background.
7. **Click Record - Analog - Start Recording**	The XRunner animation is initiated. All mouse functions have started to record.
8. From the first SUN5 xterm, open the MDA user interface GUI.	MDA user interface GUI appears. The GUI has five buttons: MDA, Schedule&Configuration, Retrieve -Archive, Log Browser, and MDB Metrics. C-MSS-18340
9. On the MDA GUI, **select the Schedule&Configuration** button.	Two windows appear with Time, Task, Host and the other has CSS FTP, Process Log, Archive Log buttons. C-MSS-18340
10.**Select a Task** from the display	C-MSS-18340
11.**Select the CSS FTP Log**	The configDialogShell appears with the Task that has been selected. The following fileds appear: Host Name, Interval, Size Limit, Absolute Time, Absolute Time List, Selection. C-MSS-18340
12.**Change the Absolute Time value and update.**	The Absolute Time is updated with the new time. C-MSS-18200 C-MSS-18220
13.**Click the Add button**.	The record is added to the Configuration. C-MSS-18340 C-MSS-18260 C-MSS-18270 C-MSS-18330 C-MSS-18220
14.**Select the same Task from the display.**	C-MSS-18340
15. **Select the CSS FTP Log**	The configDialogShell appears with the Task that has been selected. The following fileds appear: Host Name, Interval, Size Limit, Absolute Time, Absolute Time List, Selection. C-MSS-18340
16.**Click the Remove button.**	The record is removed from the Configuration. C-MSS-18340 C-MSS-18260 C-MSS-18270 C-MSS-18330 C-MSS-18220
17.**Select Close to shutdown the MDA GUI.**	MDA GUI closes.
18.**From the XRunner window, Click Record - Stop Recording**.	The XRunner animation stops recording.
19.**Move the scroll bar to the top of the Xrunner window and click on the first line of the animated text.**	An arrow is pointing to the first line of text.
20.**Select Run - Run From Arrow.**	The XRunner animation is starting.
21.**Click File - Exit**	XRunner is terminated.
22.**Click File - Save As**	The XRunner animation is save in the users /home directory.
23.**Click File - Exit**	XRunner is terminated.
24.Exit window, test is completed.	

4.10.4.2 Test Case 2: Processing Log File (TC030.002)

This test case verifies the capability of the MDA service to process the management data log files while still maintaining the integrity of the management database. MDA processes this data by loading selected information to the Management RDBMS for the purposes of longer term analysis and reporting. The data MDA loads is configurable and includes Counter Metrics, Duration Metrics, Summation/Average Metrics, and Detail Events.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MCI: MsMdProcessLog, MsMdEventField, MsMdMetrics, MsMdLoadMdb

Data: XRunner script, User input data

Tools: XRunner 4.0, MDA user interface GUI

Test Input:

Inputs to this test case include management data log files.

Test Output:

Outputs to this test case include loading Counter Metrics, Duration Metrics, Summation/Average Metrics, and Detail Events data to the Management RDBMS.

Success Criteria:

This test is successful when the MDA processes the management data log files and loads these data to the Management RDBMS.

Test Procedures:

Test Case ID: TC030.002	
Test Name:Processing Log File	
Test Steps:	Comments:
1. Log onto the MSS SUN5 Server	This test can also be done on the MSS HP Server. Note: At the present time, there isn't a n HP XRunner license.
2. Set the Display environment: **setenv DISPLAY *<host ip address>:0.0***	The environment Display is to ensure that all responses be returned to the machine screen that the user logs onto.
3. Change to directory structure where the software resides: **cd (for example: *cd /usr/testa*)**	
4. Bring up another MSS SUN5 Server xterm.	
5. Set the Display environment: **setenv DISPLAY *<host ip address>:0.0***	

6. Start the Xrunner application: **xrun &**	The XRunner application will run in the background.
7. **Click Record - Analog - Start Recording**	The XRunner animation is initiated. All mouse functions have started to record.
8. **From the first SUN5 xterm, open the MDA user interface GUI.**	MDA user interface GUI appears. The GUI has five buttons: MDA, Schedule&Configuration, Retrieve -Archive, Log Browser, and MDB Metrics. C-MSS-18340 C-MSS-18050
9. **On the MDA GUI, select the Log Browser button.**	The Log Browser window appears.. C-MSS-18340 C-MSS-18050 C-MSS-18070
10.**Click Select pulldown menu for processing.**	When Select pulldown menu is clicked, three cascade pulldown menus of single select filters (by Category, Priority, Subsystem,) is displayed with the Custom Select option. C-MSS-18340 C-MSS-18070 C-MSS-18340
11.**Click 'Category'**	When 'Category' is selected, five cascade pulldown submenus are displayed: Fault, Performance, Security Alert, Topology Change, and Transaction. When one of the options is selected, all events of the selected option is displayed in the events box. C-MSS-18340
12.**Click 'Priority'**	When 'Priority' is selected, four cascade pulldown submenus are displayed: Low, Medium Low, Medium, and High. When one of the options is selected, all events of the selected option is displayed in the events box. C-MSS-18340
13.**Click 'Subsystem'**	When 'Subsystem' is selected, eight cascade pulldown submenus are displayed: CSS, DMS, DPS, DSS, INS, IOS, MSS, and Pdps. When one of the options is selected, all events of the selected option is displayed in the events box. C-MSS-18340
14.**From Select pulldown menu, click 'Custom Select' or select the 'Custom Select' button at the bottom of the ECS Log Browser window.**	When the 'Custom Select' is clicked, the custom select window is displayed. (Currently Category, Priority, Subsystem, and Type are available for testing. but ApplicationID, CSCI, InstanceID, Mode, ProcessID, and ProgramID functionally will be provided in Phase 3. C-MSS-18340
15.**Click cancel to close window**.	
16. **Click Sort pulldown menu or Select pulldown menu for processing**.	If Sort pulldown menu is clicked, six choices of single sort and a Custom Sort for multi-level sorting will appear: Time Stamp, Category, Type, Priority, Subsystem, CSCI. C-MSS-18340

17.**Click 'Time Stamp'**	All events will be listed in chronological order. Note: This process will take a few minutes to generate due to the size of mss.log. C-MSS-18340
18.**Click 'Category'**	All events will be grouped in categories. Note: This process will take a few minutes to generate due to the size of mss.log. C-MSS-18340
19.**Click 'Type'**	All events with same types will be grouped and listed. Note: This process will take a few minutes to generate due to the size of mss.log. C-MSS-18340
20.**Click 'Priority'**	All events with the same priorities will be grouped and listed. Note: This process will take a few minutes to generate due to the size of mss.log. C-MSS-18340
21.**Click 'Subsystem'**	All events will be grouped by susbsystems and listed. Note: This process will take a few minutes to generate due to the size of mss.log. C-MSS-18340
22.**Click 'CSCI'**	All events will be grouped by CSCI and listed. Note: This process will take a few minutes to generate due to the size of mss.log. C-MSS-18340
23.**From the Sort pulldown menu, click 'Custom Sort' or click the 'Custom Sort' button** at the bottom of the ECS Log Browser window.	When the Custom Sort is selected, the Custom Sort window appears and it gives the user the option to select multi-level of sorting in forward and reverse order of the following: Time Stamp, Category, Type, Priority, Subsystem, and CSCI. C-MSS-18340
24.**If Edit pulldown menu is clicked,** two choices of undo and reset will appear.	Undo = go back once reset = go back to the original (1st list)
25.**Select File - Exit**	The ECS Log Browser windows closes.
26.On the Xrunner window, **select Record - Stop Recording.**	This stops the animation that was running.
27.**Select File - Save As**	This saves the animation to an XRunner script in the user's /home directory.
28.From the XRunner window, **click Record - Stop Recording.**	The XRunner animation stops recording.
29.Move the scroll bar to the top of the Xrunner window and click on the first line of the animated text.	An arrow is pointing to the first line of text.
30.**Select Run - Run From Arrow**.	The XRunner animation is starting.
31.**Click File - Exit**	XRunner is terminated.
32.**Click File - Save As**	The XRunner animation is save in the users /home directory.
33.**Click File - Exit**	XRunner is terminated.
34.Exit window, test is completed.	

4.10.4.3 Test Case 3: Archive Management Log Files (TC030.003)

This test case verifies the capability of the MDA service to schedule the archival of management log files and transfer management data at the DAACs site to the SMC.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MCI: MsMdManager, MsMdConfigurationEntry, MsMdConfigurationList

Data: XRunner script, User input data

Tools: XRunner 4.0, MDA user interface GUI

Test Input:

Inputs to this test case include management data log files.

Test Output:

Outputs to this test case include log files archived at the DAAC sites and at the SMC.

Success Criteria:

This test is successful when the MDA can transfer and archive the log files to the ECS Data Archives.

Test Procedures:

Test Case ID: TC030.003	
Test Name:Archive Management Log Files	
Test Steps:	Comments:
1. Log onto the MSS SUN5 Server	This test can also be done on the MSS HP Server. Note: At the present time, there isn't a n HP XRunner license.
2. Set the Display environment: **setenv DISPLAY *<host ip address>:0.0***	The environment Display is to ensure that all responses be returned to the machine screen that the user logs onto.
3. Change to directory structure where the software resides: **cd (for example: *cd /usr/testa*)**	
4. Bring up another MSS SUN5 Server xterm	
5. Set the Display environment: **setenv DISPLAY *<host ip address>:0.0***	
6. Start the Xrunner application: **xrun &**	The XRunner application will run in the background.
7. **Click Record - Analog - Start Recording**	The XRunner animation is initiated. All mouse functions have started to record.
8. From the first SUN5 xterm, **open the MDA user interface GUI.**	MDA user interface GUI appears. The GUI has five buttons: MDA, Schedule&Configuration, Retrieve -Archive, Log Browser, and MDB Metrics. C-MSS-18340 C-MSS-18060 C-MSS-18070 C-MSS-18040

9. **Select the Retrieve - Archive button.**	The Retrieve - Archive dialog box appears with Host Name, a time option for Specific Time and Range Time, Start Time, Stop Time. C-MSS-18340 C-MSS-18260 C-MSS-18270
10.**Complete the following fields:** ***Host Name*** **(the machine you are on**) ***Specific Time*** ***Start Time*** ***Stop Time*** ***Click Submit***	The data is archived for the time period entered. Note the date and time ranges that is inputted. C-MSS-18340 C-MSS-18260 C-MSS-18270 C-MSS-18060 C-MSS-18070 C-MSS-18350 C-MSS-18220 C-MSS-18070 C-MSS-18040
11.**On the Xrunner window, select Record - Stop Recording**.	This stops the animation that was running.
12.**Select File - Save As**	This saves the animation to an XRunner script in the user's /home directory.
13.**From the XRunner window, Click Record - Stop Recording.**	The XRunner animation stops recording.
14.**Move the scroll bar to the top of the Xrunner window and click on the first line of the animated text.**	An arrow is pointing to the first line of text.
15.**Select Run - Run From Arrow.**	The XRunner animation is starting.
16.**Click File - Exit**	XRunner is terminated.
17.**Click File - Save As**	The XRunner animation is save in the users /home directory.
18.**Click File - Exit**	XRunner is terminated.
19.Exit window, test is completed.	

4.10.4.4 Test Case 4: Retrieve Management Log Files (TC030.004)

This test case verifies the capability of the Management Data Access Service to schedule the retrieval of management log files from the DAAC's.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MsMdManager, MsMdConfigurationEntry, MsMdConfigurationList

Data: XRunner script, User input data

Tools: XRunner 4.0, MDA user interface GUI

Test Input:

Inputs to this test case include keyins to open and read data from the MDA user interface GUI.

Test Output:

Outputs to this test case include new configuration data and an XRunner script.

Success Criteria:

This test is successful when the MDA user interface containing information on host events can be retrieved into the management database. The successful creation and execution of an XRunner script is also required.

Test Procedures:

Test Case ID: TC030.004	
Test Name: Retrieve Management Log Files	
Test Steps:	Comments:
1. Log onto the MSS SUN5 Server	This test can also be done on the MSS HP Server. Note: At the present time, there isn't a n HP XRunner license.
2. Set the Display environment: **setenv DISPLAY *<host ip address>:0.0***	The environment Display is to ensure that all responses be returned to the machine screen that the user logs onto.
3. Change to directory structure where the software resides: cd **(for example: *cd /usr/testa*)**	
4. Bring up another MSS SUN5 Server xterm	
5. Set the Display environment: **setenv DISPLAY *<host ip address>:0.0***	
6. Start the Xrunner application: **xrun &**	The XRunner application will run in the background.
7. **Click Record - Analog - Start Recording**	The XRunner animation is initiated. All mouse functions have started to record.
8. **From the first SUN5 xterm, open the MDA user interface GUI.**	MDA user interface GUI appears. The GUI has five buttons: MDA, Schedule&Configuration, Retrieve -Archive, Log Browser, and MDB Metrics. C-MSS-18340 C-MSS-18040 C-MSS-18060 C-MSS-18070 C-MSS-18260
9. **Select the Retrieve - Archive button**.	The Retrieve - Archive dialog box appears with Host Name, a time option for Specific Time and Range Time, Start Time, Stop Time. C-MSS-18340 C-MSS-18260 C-MSS-18270
10.**Select Retrieve**	The Retrieve window appears. C-MSS-18340

11.Complete the following fields: ***Host Name (*****the machine you are on*****)*** ***Specific Time*** ***Start Time*** ***Stop Time*** ***Click Submit***	The data is retrieved for the time period entered. Note the date and time ranges that is inputted. C-MSS-18340 C-MSS-18260 C-MSS-18270 C-MSS-18060 C-MSS-18070 C-MSS-18350 C-MSS-18200 C-MSS-18220 C-MSS-18350 C-MSS-18280
12.**On the Xrunner window, select Record - Stop Recording.**	This stops the animation that was running.
13.**Select File - Save As**	This saves the animation to an XRunner script in the user's /home directory.
14.**From the XRunner window, Click Record - Stop Recording**.	The XRunner animation stops recording.
15.***Move the scroll bar to the top of the XRunner window and click on the first line of the animated text.***	An arrow is pointing to the first line of text.
16.**Select Run - Run From Arrow**.	The XRunner animation is starting.
17.**Click File - Exit**	XRunner is terminated.
18.**Click File - Save As**	The XRunner animation is save in the users /home directory.
19.**Click File - Exit**	XRunner is terminated.
20.Exit window, test is completed.	

4.10.4.5 Test Case 5: Performance Data Extraction (TC030.005)

This test case demonstrates the ability to extract performance data from the log files and schedule the transfer and loading of the performance data from the log files into the management database at the site. Specifically, this test ensures that the MSS Performance Management Application Service is capable of:

- calculating the statistics (MDT, MTBM, MTTR) for the purpose of supporting RMA analysis for managed objects.
- retaining the calculated RMA statistics in a repository accessible for further analysis.
- retaining the calculated RMA statistics in a repository accessible for further analysis.
- extracting summarized site status information from logged performance data.
- extracting values of performance metrics gathered for a specified managed object.
- generating a graph of the extracted performance metric values.

<u>Test Configuration:</u>

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MsMdManager, Mgmt Database, MsPmManager

Data: XRunner script, User input data

Tools: XRunner 4.0, MDA user interfaceGUI

Test Input:

Log into the management data log file on each managed host.

Test Output:

The expected results of this test include the ability to extract performance data from the management data log file.

Success Criteria:

This test is successful when all selected performance data has been properly extracted from the management data log file.

Test Procedures:

Test Case ID: TC030.005	
Test Name: Performance Data Extraction	
Test Steps:	Comments:
1. Log onto the MSS SUN5 Server.	
2. Set the Display environment: **setenv DISPLAY *<host ip address>:0.0***	This test can also be done on the MSS HP Server. Note: At the present time, there isn't a n HP XRunner license.
3. Change to directory structure where the software resides: **cd (for example: *cd /usr/testa*)**	
4. Bring up another MSS SUN5 Server xterm	
5. Set the Display environment: **setenv DISPLAY *<host ip address>:0.0***	
6. Start the XRunner application: **xrun &**	The XRunner application will run in the background.
7. **Click Record - Analog - Start Recording**	The XRunner animation is initiated. All mouse functions have started to record.
8. Demonstrate the capability to extract the performance data from the management data log file.	
9. **From the first SUN5 xterm, open the MDA user interface GUI.**	MDA user interface GUI appears. The GUI has five buttons: MDA, Schedule &Configuration, Retrieve - Archive, Log Browser, and MDB Metrics. C-MSS-18340 C-MSS-18040 C-MSS-18060 C-MSS-18070 C-MSS-18260

10. Initialize the MDA user interface GUI to calculate the ***performance*** statistics.	The MDA user interface for performance is initialized. C-MSS-66135 C-MSS-66137
11. ***Select*** **the MTBM (*Mean Time Between Maintenance*) button.**	MTBM selections appear. C-MSS-66135 C-MSS-66137
12. ***Select and double click on one.***	The MTBM window opens and displays MTBM attributes. C-MSS-66135 C-MSS-66137
13.Verify that the correct data is in the proper fields.	
14.***Save this file*** to verify that records can be appended to a log file.	The MTBM file is saved in the management database. C-MSS-66135 C-MSS-66137
15. Open the saved file, it should be in a graph format.	File opens and a graph of MTBM is displayed.
16.***Select*** **the MTTR (*Mean Time to Repair*) button.**	MTTR selections appear. C-MSS-66135 C-MSS-66137
17.***Select and double click on one.***	The MTTR window opens and displays MTTR attributes. C-MSS-66135 C-MSS-66137
18.Verify that the correct data is in the proper fields.	
19. ***Save this file*** to verify that records can be appended to a log file.	The MTTR file is saved in the management database. C-MSS-66135 C-MSS-66137
20.Open the saved file, it should be in a graph format.	File opens and a graph of MTTR is displayed.
21.***Select*** **the MTBCM (*Mean Time Between Corrective Maintenance*) button.**	MTBCM selections appear. C-MSS-66135 C-MSS-66137
22. ***Select and double click on one.***	The MTBCM window opens and displays MTBCM attributes. C-MSS-66135 C-MSS-66137
23.Verify that the correct data is in the proper fields.	
24.***Save this file*** to verify that records can be appended to a log file.	The MTBCM file is saved in the management database. C-MSS-66135 C-MSS-66137
25.Open the saved file, it should be in a graph format.	File opens and a graph of MTBCM is displayed.
26.***Select*** **the MTBPM (*Mean Time Between Preventive Maintenance*) button.**	MTBPM selections appear. C-MSS-66135 C-MSS-66137

27.***Select and double click on one.***	The MTBPM window opens and displays MTBCM attributes. C-MSS-66135 C-MSS-66137
28.Verify that the correct data is in the proper fields.	
29.***Save this file*** to verify that records can be appended to a log file.	The MTBPM file is saved in the management database. C-MSS-66135 C-MSS-66137
30. Open the saved file, it should be in a graph format.	File opens and a graph of MTBPM is displayed.
31.***Select*** **the MDT (*Mean Down Time)* button.**	MDT selections appear. C-MSS-66135 C-MSS-66137
28.***Select and double click on one.***	The MDT window opens and displays MDT attributes. C-MSS-66135 C-MSS-66137
29.Verify that the correct data is in the proper fields.	
30.***Save this file*** to verify that records can be appended to a log file.	The MDT file is saved in the management database. C-MSS-66135 C-MSS-66137
31.Open the saved file, it should be in a graph format.	File opens and a graph of MDT is displayed.
32. Exit the MDA user interface GUI.	The MDA user interface GUI window closes.
33.On the XRunner window, **select Record - Stop Recording.**	This stops the animation that was running.
34.**Select File - Save As**	This saves the animation to an XRunner script in the user's /home directory.
35.From the XRunner window, **click Record - Stop Recording.**	The XRunner animation stops recording.
36.Move the scroll bar to the top of the XRunner window and click on the first line of the animated text.	An arrow is pointing to the first line of text.
37.**Select Run - Run From Arrow**.	The XRunner animation is starting.
38.**Click File - Exit**	XRunner is terminated.
39.**Click File - Save As**	The XRunner animation is save in the users /home directory.
40.**Click File - Exit**	XRunner is terminated.
41.Exit window, test is completed.	

4.10.4.6 Test Case 6: Fault Data Extraction (TC030.006)

This test case demonstrates the ability to extract fault data from the log files and to schedule the transfer and loading of the fault data from the log files into the management database at the site. Specifically, this test ensures that the MSS Performance Management Application Service is capable of:

- calculating the statistics (MDT, MTBM, MTTR) for the purpose of supporting RMA analysis for managed objects.

- retaining the calculated RMA statistics in a repository accessible for further analysis.
- extracting summarized site status information from logged performance data.
- extracting values of performance metrics gathered for a specified managed objects.
- generating a graph of the extracted performance metric values.

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MsMdManager, Mgmt Database, MsPmManager

Data: XRunner script, User input data

Tools: XRunner 4.0, MDA GUI

Test Input:

Log into the management data log file on each managed host.

Test Output:

The expected results of this test include the ability to extract fault data from the management data log file.

Success Criteria:

This test is successful when all fault data has been properly extracted from the management data log file.

Test Procedures:

Test Case ID: TC030.006	
Test Name: Fault Data Extraction	
Test Steps:	Comments:
1. Log onto the MSS SUN5 Server.	
2. Set the Display environment: **setenv DISPLAY *<host ip address>:0.0***	This test can also be done on the MSS HP Server. Note: At the present time, there isn't a n HP XRunner license.
3. Change to directory structure where the software resides: **cd (for example: *cd /usr/testa*)**	
4. Bring up another MSS SUN5 Server xterm	
5. Set the Display environment: **setenv DISPLAY *<host ip address>:0.0***	
6. Start the XRunner application: **xrun &**	The XRunner application will run in the background.
7. **Click Record - Analog - Start Recording**	The XRunner animation is initiated. All mouse functions have started to record.

8. Demonstrate the capability to extract the performance data from the management data log file.	
9. **From the first SUN5 xterm, open the MDA user interface GUI.**	MDA user interface GUI appears. The GUI has five buttons: MDA, Schedule &Configuration, Retrieve -Archive, Log Browser, and MDB Metrics. C-MSS-18340 C-MSS-18040 C-MSS-18060 C-MSS-18070 C-MSS-18260
10. Initialize the MDA user interface GUI to calculate the ***fault*** statistics.	The MDA user interface for fault is initialized. C-MSS-66135 C-MSS-66137
11. ***Select*** **the MTBM (*Mean Time Between Maintenance*) button.**	MTBM selections appear. C-MSS-66135 C-MSS-66137
12. ***Select and double click on one.***	The MTBM window opens and displays MTBM attributes. C-MSS-66135 C-MSS-66137
13.Verify that the correct data is in the proper fields.	
14.***Save this file*** to verify that records can be appended to a log file.	The MTBM file is saved in the management database. C-MSS-66135 C-MSS-66137
15. Open the saved file, it should be in a graph format.	File opens and a graph of MTBM is displayed.
16.***Select*** **the MTTR (*Mean Time to Repair*) button.**	MTTR selections appear. C-MSS-66135 C-MSS-66137
17.***Select and double click on one.***	The MTTR window opens and displays MTTR attributes. C-MSS-66135 C-MSS-66137
18.Verify that the correct data is in the proper fields.	
19. ***Save this file*** to verify that records can be appended to a log file.	The MTTR file is saved in the management database. C-MSS-66135 C-MSS-66137
20.Open the saved file, it should be in a graph format.	File opens and a graph of MTTR is displayed.
21.***Select*** **the MTBCM (*Mean Time Between Corrective Maintenance*) button.**	MTBCM selections appear. C-MSS-66135 C-MSS-66137
22. ***Select and double click on one.***	The MTBCM window opens and displays MTBCM attributes. C-MSS-66135 C-MSS-66137
23.Verify that the correct data is in the proper fields.	
24.***Save this file*** to verify that records can be appended to a log file.	The MTBCM file is saved in the management database. C-MSS-66135 C-MSS-66137

25.Open the saved file, it should be in a graph format.	File opens and a graph of MTBCM is displayed.
26.***Select*** **the MTBPM (*Mean Time Between Preventive Maintenance*) button.**	MTBPM selections appear. C-MSS-66135 C-MSS-66137
27.***Select and double click on one.***	The MTBPM window opens and displays MTBCM attributes. C-MSS-66135 C-MSS-66137
28.Verify that the correct data is in the proper fields.	
29.***Save this file*** to verify that records can be appended to a log file.	The MTBPM file is saved in the management database. C-MSS-66135 C-MSS-66137
30. Open the saved file, it should be in a graph format.	File opens and a graph of MTBPM is displayed.
31.***Select*** **the MDT (*Mean Down Time)* button.**	MDT selections appear. C-MSS-66135 C-MSS-66137
28.***Select and double click on one.***	The MDT window opens and displays MDT attributes. C-MSS-66135 C-MSS-66137
29.Verify that the correct data is in the proper fields.	
30.***Save this file*** to verify that records can be appended to a log file.	The MDT file is saved in the management database. C-MSS-66135 C-MSS-66137
31.Open the saved file, it should be in a graph format.	File opens and a graph of MDT is displayed.
32. Exit the MDA user interface GUI.	The MDA user interface GUI window closes.
33.On the XRunner window, **select Record - Stop Recording.**	This stops the animation that was running.
34.**Select File - Save As**	This saves the animation to an XRunner script in the user's /home directory.
35.From the XRunner window, **click Record - Stop Recording.**	The XRunner animation stops recording.
36.Move the scroll bar to the top of the XRunner window and click on the first line of the animated text.	An arrow is pointing to the first line of text.
37.**Select Run - Run From Arrow**.	The XRunner animation is starting.
38.**Click File - Exit**	XRunner is terminated.
39.**Click File - Save As**	The XRunner animation is save in the users /home directory.
40.**Click File - Exit**	XRunner is terminated.
41.Exit window, test is completed.	

4.10.5 Baseline Manager Thread (TC031)

The Baseline Manager Service provides the capability to record and track the resources within an ECS baseline. This Baseline Manager identifies what is contained within a released baseline, updates the status of its components, and traces the baseline to requirements. It also catalogs configuration items, both hardware and software, records and identifies documentation for each

site configuration, and maintains a history of changes to individual resources and to system and site configurations.

4.10.5.1 Test Case 1: Maintenance of Configuration Items (TC031.001)

This test case demonstrates the ability of the Baseline Manager to record and maintain configuration items including ECS deployed hardware and software and SCF-provided configuration data.

Test Configuration:

Hardware: LSM and SMC Workstation

Software: Baseline Manager

Data: ECS baseline information

Tools: N/A

Test Input:

Inputs to this test case include application commands to enter and display configuration item information.

Test Output:

Output includes stored configuration/baseline records.

Success Criteria:

This test is successful when the configuration information is retrievable from the database and contains all user entered information.

Test Procedures:

Test Case ID: TC031.001	
Test Name: Maintenance of Configuration Items	
Test Steps:	Comments:
1. Log into each LSM and SMC workstation and execute Baseline Manager application. **xrp**	Baseline Manager application will appear.
2. Execute commands to enter ECS baseline information.	Baseline information includes hardware, software, documentation, and science algorithm items.
3. Verify the baseline manager tracks the following: a. ECS subsystems, networks, and configured system and network devices such as workstations, servers, and routers. b. ECS releases and site baselines c. ECS hardware and software resources designated as configuration items.	C-MSS-40000, C-MSS-40120 SMC-2510 (partial), SMC-2515 (partial), SMC-6340 (partial)

d. specifications associated with configuration items e. technical documentation and test materials f. scientific algorithms, including software, data and test materials	
4. Verify the baseline manager tracks and maintains SCF-provided configuration data including: a. algorithm development version numbers, identification codes and reference numbers b. SCF point of contact's name and organization. c. associated files' names, formats, sized, and descriptions d. number of files by category and type.	C-MSS-40100 SMC-2510 (partial), SMC-2515 (partial) PGS-0950
5. Verify the baseline manager tracks and maintains software-critical and security sensitive items.	C-MSS-40240 SMC-2510 (partial)
5. Submit information into baseline manager to store and update.	Stores all baseline information into database.
6. Exit baseline manager.	

4.10.5.2 Test Case 2: Version Control Test (TC031.002)

This test verifies that the baseline manager maintains a version history of configuration controlled resources according to each site's operational baseline.

Test Configuration:

Hardware: LSM Workstation

Software: Baseline Manager

Data: Previous version history

Tools: N/A

Test Input:

Inputs to this test case include application commands to recall and display baseline resources and their historical data.

Test Output:

Output includes version history of resources displayed to screen.

Success Criteria:

This test is successful when the user is able to recall version histories of each of the baseline resources from the baseline manager.

Test Procedures:

Test Case ID: TC031.002	
Test Name: Version Control Test	
Test Steps:	Comments:
1. Log into each LSM and SMC workstation and execute Baseline Manager application. **xrp**	Baseline Manager application will appear.
2. Access each type of configuration item through their interface screens and verify the baseline manager contains a version history.	C-MSS-40010 SMC-6345 (partial)
3. Exit baseline manager application.	

4.10.5.3 Test Case 3: Forwarding Baseline Records to SMC (TC031.003)

This test verifies the capability of a local site baseline manager to forward records that describe each site's operation baseline.

Test Configuration:

Hardware: LSM and SMC Workstation

Software: Baseline Manager

Data: Existing LSM baseline information within Baseline Manager

Tools: N/A

Test Input:

Inputs to this test case include application commands to recall a site's operation baseline, and package it for forwarding to the SMC.

Test Output:

Outputs to this test include transfer confirmation messages and configuration record lists.

Success Criteria:

This test is successful when the SMC baseline manager is able to retrieve and store each site's operation baseline.

Test Procedures:

Test Case ID: TC031.003	
Test Name: Forwarding Baseline Records to SMC	
Test Steps:	Comments:
1. Log into each LSM and SMC workstation and execute Baseline Manager application. **xrp**	Baseline Manager application will appear.

2. On the LSM workstation identify and access all site operational baseline and the version and status of configuration controlled resources related to the baseline. Submit this information for transfer to the SMC.	C-MSS-40030 SMC-2515 (partial)
3. On the LSM workstation identify a "level of assembly" list that describes the composition of configuration items and submit this information for transfer to the SMC.	C-MSS-40040 SMC-2515 (partial)
4. Transfer all information to the SMC.	Records and lists specific to the LSM will be transferred to the SMC baseline manager.
5. From the SMC baseline manager, verify all information from LSM baseline manager was transferred and stored to SMC database.	Verification that records and lists from the LSM were transferred and stored to the SMC baseline manager. All information will be analyzed for completeness.
6. From the SMC baseline manager, verify that ECS resources are characterized as system-wide or site-wide.	C-MSS-40280
6. Exit both the LSM and SMC baseline managers.	

4.10.5.4 Test Case 4: Record Maintenance Test (TC031.004)

This test verifies that the baseline manager maintains records of configuration items at each site and the SMC. Records on configuration items include status, dependency traceability, and component structure.

Test Configuration:

Hardware: LSM and SMC workstation

Software: Baseline Manager

Data: LSM and SMC baseline information

Tools: N/A

Test Input:

Inputs to this test case include application commands to recall and view site baseline information.

Test Output:

Outputs to this test case include reports showing information based on configuration items and user requested reports.

Success Criteria:

This test is successful when the baseline manager is able to generate reports based on query information from the user.

Test Procedures:

Test Case ID: TC031.004	
Test Name: Record Maintenance Test	
Test Steps:	Comments:
1. Log into each LSM and SMC workstation and execute Baseline Manager application. **Xrp**	Baseline Manager application will appear.
2. Access and view each configuration item stored within the baseline manager. Verify for each configuration item, the baseline manager records: a. the current version; b. current version's specifications and technical, operations, and maintenance documentation c. specification and technical documentation history d. "level" of assembly" representation of the components e. version history	C-MSS-40060, C-MSS-40200 SMC-2515 (partial), SMC-2510 (partial)
3. Verify that for each configuration item, the baseline manager has recorded its object dependencies.	C-MSS-40080 The user will analyze the reported dependencies against the "real-time" dependencies to test for completeness.
4. Verify that the baseline manager, for each site, maintains traceability among operational baselines and releases. Using the baseline manager application the user will link configuration items to baselines and releases.	C-MSS-40070 SMC-2515 (partial) Creating links between configuration items and baselines/releases is a test of the traceability of the tool.
5. Submit the link information to the database and perform a query to access and view the traceability information.	Verifies the baseline manager stored the traceability information to the database and allows users to view this information.
6. Perform a query on each configuration item, and verify the baseline manager lists all components related to that item.	C-MSS-40110 SMC-2510 (partial), SMC-2515 (partial)
7. Perform a query for all configuration items related to the current ECS release. Verify the baseline manager maintains historical status records including: a. latest baseline plus approved changes b. baseline history c. latest release documentation d. "level of assembly" representation of the subsystem and configuration item versions that comprise the rele4ase configuration e. history of changes, including changes to subordinate units/components. f. effectivitiy and installation status at operational sites g. release configuration	C-MSS-40210 SMC-2510 (partial)
8. Exit baseline manager.	

4.10.5.5 Test Case 5: Distributed Site Information Test (TC031.005)

This test verifies that the baseline manager at the SMC maintains information related to each sites baseline and distributes this information system-wide.

Test Configuration:

Hardware: LSM and SMC workstation

Software: Baseline Manager

Data: LSM and SMC baseline information

Tools: N/A

Test Input:

Inputs to this test case include application commands that will allow a user at the LSM baseline manager, view information about other site baselines from the SMC baseline manager.

Test Output:

Outputs to this test include site specific reports/information, viewed from the baseline manager.

Success Criteria:

This test is successful when the SMC baseline manager distributes site specific information system-wide to each LSM baseline manager.

Test Procedures:

Test Case ID: TC031.005	
Test Name: Distributed Site Information Test	
Test Steps:	Comments:
1. Log into each LSM and SMC workstation and execute Baseline Manager application. **Xrp**	Baseline Manager application will appear.
2. Perform a query on a configuration item at the LSM, and record all the information related to that configuration item, including version and operational status.	
3. Perform a query on the same configuration item at the SMC and verify the contents of the record match that of the configuration item from the LSM.	C-MSS-40140 SMC-2510 (partial), SMC-6340 (partial) Verifies the transfer of information from the LSM to the SMC baseline manager succeeded, and that the SMC baseline manager distributes this information out to all sites. (In this case the LSM)
4. From the SMC baseline manager, create a query that will extract the current version of ECS resources (hardware and software) that were deployed to the LSM.	Verifies that the SMC baseline manager records and tracks baseline information for each site.
5. From the LSM baseline manager, create and submit the same query and verify all information is correct.	Verifies the SMC baseline manager tracks and maintains current information about each LSM.
6. From the LSM baseline manager, create and submit the same query to the SMC baseline manager.	C-MSS-40150 SMC-2515 (partial) Verifies the SMC baseline manager distributes baseline information to all sites.

7. From the SMC baseline manager, create a query that will extract previous versions of ECS resources (hardware and software) that were deployed to the LSM.	Verifies that the SMC baseline manager records and tracks all baseline information for each site.
8. From the LSM baseline manager, create and submit the same query and verify all information is correct.	Verifies the SMC baseline manager tracks and maintains current information about each LSM.
9. From the LSM baseline manager, create and submit the same query to the SMC baseline manager.	C-MSS-40150 SMC-2515 (partial) Verifies the SMC baseline manager distributes baseline information to all sites.
10. From the SMC baseline manager, create and submit a query that identifies all documentation associated with versions of ECS resources deployed to the LSM.	C-MSS-40160 SMC-2510 (partial) Verifies the SMC baseline manager records and tracks baseline information for each site.
11. From the LSM baseline manager, create and submit the same query and verify all information is correct.	Verifies the SMC baseline manager tracks and maintains current information about each LSM.
12. From the LSM baseline manager, create and submit the same query to the SMC baseline manager.	C-MSS-40180, C-MSS-40250 SMC-2510 (partial) Verifies the SMC baseline manager distributes baseline information to all sites.
13. From the SMC baseline manager, create and submit a query that identifies the change status ECS resources deployed to the LSM.	Verifies the SMC baseline manager records and tracks baseline information for each site.
14. From the LSM baseline manager, create and submit the same query and verify all information is correct.	Verifies the SMC baseline manager tracks and maintains current information about each LSM.
15. From the LSM baseline manager, create and submit the same query to the SMC baseline manager.	C-MSS-40260 SMC-2510 (partial) Verifies the SMC baseline manager distributes baseline information to all sites.
16. From the SMC baseline manager, create and submit a query that identifies all resources comprising ECS baselines and releases.	Verifies the SMC baseline manager records and tracks baseline information.
18. From the LSM baseline manager, create and submit the same query to the SMC baseline manager.	C-MSS-40270 SMC-2510 (partial) Verifies the SMC baseline manager distributes baseline information to all sites.
19. Exit both baseline managers.	

4.10.5.6 Test Case 6: Distributed Baseline Changes Test (TC031.006)

This test verifies the baseline manager at the SMC will maintain and distribute to each site, baseline changes related to ECS resources deployed at each site.

Test Configuration:

Hardware: LSM and SMC workstation

Software: Baseline Manager

Data: ECS baseline information

Tools: N/A

Test Input:

Inputs to this test case include application commands to the baseline manager to request and view baseline information.

Test Output:

Ouputs to this test case include baseline information for the site.

Success Criteria:

This test is successful when the baseline information viewed matches any changes that were recorded from the SMC baseline manager.

Test Procedures:

Test Case ID: TC031.006	
Test Name: Distributed Baseline Changes Test	
Test Steps:	Comments:
1. Log into each LSM and SMC workstation and execute Baseline Manager application. **Xrp**	Baseline Manager application will appear.
2. From the SMC baseline manager, execute commands to enter change request information into the baseline manager. The change request will be related to hardware and software changes to the ECS baseline.	Change request information will indicate enhancements or corrections to new versions of ECS resources.
3. Log into the LSM baseline manager, create and submit a query about ECS resources to the SMC baseline manager.	C-MSS-40170, C-MSS-40190 SMC-2510 (partial) Verifies the SMC records and distributes baseline change request information related to new versions of deployed ECS resources.
4. Verify the information extracted from the LSM query to that entered at the SMC	Verification for completeness.
5. Verify, upon entering change request information to ECS baselines, the baseline manager records: a. sites affected b. installation dates c. installation status	C-MSS-40220 SMC-2510 (partial) Verification that the baseline manager records and maintains ECS baseline changes.
6. Exit both baseline managers.	

4.10.5.7 Test Case 7: Baseline Information Entry Test (TC031.007)

This test verifies the baseline manager application is able to accept and store information from a user interface, or from formatted data files.

Test Configuration:

Hardware: LSM and SMC workstation

Software: Baseline Manager

Data: ECS baseline information for manual data entry and also in data files.

Tools: N/A

Test Input:

Inputs to this test include ECS baseline information to be entered into the baseline manager application. This data will be entered manually and from data files.

Test Outputs:

Outputs to this test include confirmation of data ingest, and reports verifying data was entered correctly.

Success Criteria:

This test is successful when the baseline manager accepts data entry from both a user interface and data files, and stores all information.

Test Procedures:

Test Case ID: TC031.007	
Test Name: Baseline Information Entry Test	
Test Steps:	Comments:
1. Log into each LSM and SMC workstation and execute Baseline Manager application. **Xrp**	Baseline Manager application will appear.
2. Via user interface, enter ECS baseline information into the baseline manager.	C-MSS-40290 Perform this step on both baseline managers (LSM and SMC). Verification the baseline manager accepts data entry through user interaction.
3. Create a data file that contains ECS baseline information.	Prepare data in format readable by the baseline manager.
4. Ingest data from data files (step 3) into the baseline manager.	C-MSS-40290 Perform this step on both baseline managers (LSM and SMC). Information will be ingested into baseline manager via data file.
5. At each site, create and submit a query that will extract all information entered, through user interface, and data file.	Verification that data entered and stored within baseline manager.
6. Exit both baseline managers.	

4.10.5.8 Test Case 8: Data File Production Test (TC031.008)

This test verifies the baseline manager application is able to produce output data files containing information from user queries.

Test Configuration:

Hardware: LSM and SMC workstation

Software: Baseline Manager

Data: Existing ECS baseline information

Tools: N/A

Test Input:

Inputs to this test include application commands that aid the user in viewing ECS baseline information and storing this information to a file.

Test Output:

Outputs to this test include data files containing ECS baseline information.

Success Criteria:

This test is successful when the baseline manager application produces data files which contain user requested baseline information.

Test Procedures:

Test Case ID: TC031.008	
Test Steps: Data File Production Test	Comments:
1. Log into each LSM and SMC workstation and execute Baseline Manager application. **Xrp**	Baseline Manager application will appear.
2. From each site, create and submit a query that extracts baseline management data.	
3. Through the baseline manager application, output this information to a file.	Stores contents of query to output file.
4. Verify the output data file contains all information from query.	C-MSS-40300
5. Exit both baseline managers.	

4.10.5.9 Test Case 9: Baseline Management Log Test (TC031.09)

This test verifies the baseline manager maintains and logs information including configuration management events and user specified events.

Test Configuration:

Hardware: LSM and SMC workstation

Software: Baseline Manager

Data: Existing ECS baseline information

Tools: N/A

Test Input:

Inputs to this test include application commands to generate logs.

Test Output:

Outputs to this test include logs showing configuration management and user specified events.

Success Criteria:

This test is successful when the logs generated include specified CM and user specified events.

Test Procedures:

Test Case ID: TC031.009	
Test Name: Baseline Management Log Test	
Test Steps:	Comments:
1. Log into each LSM and SMC workstation and execute Baseline Manager application. **Xrp**	Baseline Manager application will appear.
2. From each of the sites, access and view the event/history log. Verify the baseline manager records the operation types, user access (id's), date and time, and host name information.	C-MSS-40990
3. From each of the sites, generate reports from logged events associated with: a. time frames, b. operation types, c. userids, d. hosts.	C-MSS-40995 Verifies baseline manager allows custom reports.
4. Exit both baseline managers.	

4.10.5.10 Test Case 10: Resource Planner Interaction Test (TC031.010)

This test verifies the Resource Planner is able to request resource configuration records from the Baseline Manager and receive the requested information. This test verifies the integration of the Baseline Manager with Tivoli to perform the Resource Planner request.

Test Configuration

Hardware: LSM workstation

Software: Baseline Manager, Tivoli, Resource Planner

Data: Existing resource configuration records

Tools: N/A

Test Input:

Inputs to this test include application commands from the Resource Planner to request resource configuration records.

Test Output:

Outputs to this test include a popup window on the planning workstation that includes the resource configuration record information.

Success Criteria:

This test is successful when the resource configuration record information in the popup window corresponds to the user query, and the interaction between the Resource Planner, Tivoli, and the Baseline Manager works properly.

Test Procedures:

Test Case ID: TC031.010	
Test Name: Resource Planner Interaction Test	
Test Steps:	Comments:
1. Log into the Resource Planning workstation and activate the Resource Planner.	Executes the Resource Manager application.
2. From the Resource Planner application, request resource configuration records, using the GUI dropdown menus.	Request will be sent to the Baseline Manager to perform the query, and results will return. All interaction via Tivoli.
3. When GUI popup window appears with results record them, and exit.	Information returned will be compared to information from actual Baseline Manager query.
4. Log into each LSM and SMC workstation and execute Baseline Manager application. **Xrp**	Baseline Manager application will appear.
5. From the Baseline Manager, request the identical query as in step 2.	Record information.
6. Compare results with that received from step 3, and verify information is identical.	Verification that the Resource Planner application queried and received the all information.

4.10.6 Management Services 2 Build (BC009)

The Management Services 2 build is a collection of the Software Fault Management Thread, the Security Management Thread, the Management Data Access 2 Thread, the Baseline Management Thread, the Physical Configuration Management Thread, the Trouble Ticketing Thread, the Software Change Manager Thread, and the Management Services 1 Build. The Management Services 1 Build is an aggregation of the Performance Management Thread, the Hardware Fault Management Thread, the Management Data Access 1 Thread, the Accountability Thread, the Change Request Management Thread, and the Applications Management Services Build. The Applications Management Services Build is an aggregation of the Managed Process Framework Thread, the Lifecycle Services Thread, the Discover/Monitor Thread, the MIB/Extensible Agent-Applications 2 Thread, and the External Interface Communications Build.

The objective of this test is to verify that the previously tested functionality of the threads and builds remain intact following integration.

4.10.6.1 Test Case 1: SW Fault Management Test (BC009.001)

The Integrated Software Fault Management test is a regression test of the overall functionality of the Software Fault Management Thread. To ensure that the previously tested functionality of this thread remains intact upon integration with the Management Services 2 build.

This test verifies the ability to trace faults through the system software and applications, to execute diagnostic tests, and to initiate corrective or recovery actions upon the isolation of errors in order to correct faults. This test will primarily verify software fault categories, software fault metrics, software fault attributes, and multiple sources of software faults.

Test Configuration:

Hardware: Workstations, printers, ECS standard hardware, LAN Analyzer, DCE Configured LAN

Software: HpOpenView Network Node Manager, Protocol Decoder, DCE Software/Security Server

Data: TBD

Tools: A script(s) to generate SW faults within the ECS, Monitoring and logging tool, XRunner/LoadRunner

Test Input:

Various system software, application faults and error conditions induced by the operator.

Test Output:

Expected outputs include logs that show various faults, their severity level, their categories, their fault metric information, and notification that a software fault has occurred.

Success Criteria:

This test is successful when the message log contains faults and their respective severity levels and categories and a report with software fault metric information is generated. The FMAS is able to monitor fault information from multiple sites and to notify operators of a fault within a minimum allowed time.

Test Procedures

Test Case ID: BC009.001	
Test Name: SW Fault Manager	
Test Steps:	Comments:
1. Initialize HP OpenView by entering: **ovw &**	HP OpenView B.04.00 should come up and run in the background. EOSD1703#A
2. Start up the PEER master agent.	

3. Start up the Encapsulator.	
4. Start up the SubAgent.	
5. Startup and then shutdown an application.	
6. Repeat for various failure types.	
7. Correct traps and failure notification can be detected.	DADS0901#A, PGS-0310#A, EOSD0510#A, EOSD0740#A, EOSD0750#A, EOSD0760#A, EOSD0500#A, EOSD1703#A, SMC-4305, SMC-4315, SMC-4325, SMC-4335, EOSD1710, EOSD2990, ESN-0810, ESN-0910, ESN-0920, ESN-1010, ESN-1030, ESN-0620, ESN-0690

4.10.6.2 Test Case 2: Security Management (BC009.002)

The Security Management test is a regression test of the overall functionality of the Security Management Thread. To ensure that the previously tested functionality of this thread remains intact upon integration with the Management Services 2 build.

This test verifies the ability to:

- specify privileges for authorized users
- perform periodic and scheduled security tests at the site
- receive security notification events from the sites
- detect intrusions and send notification to designate users and M&O staff
- coordinate the recovery of site compromises
- generate reports on all security violations and redirect report output to a console, printer, or disk file
- access the SMAS from a command line interface and GUI
- manage the site's authentication, authorization, and network databases

Test Configuration:

Hardware: MSS CI (local management server, management workstation), MHCI, printers, Xterm, etc., Inter networking subsystem CI (LAN Network resources), INCI

Software: HAL DCE Cell Manager, DCE ACL_EDIT, OS ACL, Router_ACL, TCP Wrapper, Tripwire, GUI S/W, MSS Management Software CI-MCI, MsScManager, MsScSMC, MScExtSys, Authentication, Authorization and Network Database

Data: TBD

Tools: TBD

Test Input:

Inputs to this test include invalid user accesses to the ECS, password audit, file system integrity check, user privileges audit, resource access control information, security notification events, site test result file, security intrusion, recovery procedures, security violation, and command line keyins for access to Authentication, Authorization, or Network Database information.

Test Output:

The expected results of this test include:

- logs that contain user group names and privileges
- security intrusion and access logs
- security violation report
- indication of access to authentication, authorization and network database information

Success Criteria:

This test is successful when:

- privileges for users can be specified and when user can be denied/allowed access to the ECS,
- site security tests can be performed,
- security intrusions are detected, logged and analyzed,
- security recovery procedures are performed via instructions, directives and tools,
- security violation reports can be generated,
- the SMAS can be accessed via GUI or the command line interface.

Test Procedures

Test Case ID: BC009.002	
Test Name: Security Management	
Test Steps:	Comments:
1) Attempt login to LSM client with invalid userid.	
2) Verify userid is denied and notification is displayed to the user.	
3) Verify login attempt is logged in the MSS audit security log.	
4) Attempt login to LSM client with valid userid and invalid password.	
5) Verify userid is denied and notification is displayed to the user.	

6) Verify login attempt is logged in the MSS audit security log.	
7) Login into LSM client with a valid userid and valid password.	Authorized LSM user login.
8) Verify user is granted access to LSM account.	NI-0470#A, NI-0480#A DADS0901#A, PGS-0310#A, EOSD0510#A, EOSD0740#A, EOSD0750#A, EOSD0760#A, EOSD0500#A, EOSD1703#A, SMC-5305, SMC-5325, SMC-5335, SMC-5345, SMC-5355, SMC-5365, SMC-6315, EOSD0040, EOSD1990, EOSD2400, EOSD2430, EOSD2480, EOSD2510, EOSD2550, EOSD2620, EOSD2640, EOSD2650, EOSD2660, EOSD2710, EOSD2990, EOSD3000, EOSD3200, ESN-1360, ESN-1365, ESN-1380, ESN-1400, ESN-1430, EOSD2100, EOSD2200

4.10.6.3 Test Case 3: Management Data Access 2 (BC009.003)

The Management Data Access test is a regression of the overall functionality of the Full Capability of Management Data Access Thread. To ensure that the previously tested functionality of this thread remains intact following integration with other threads in the Management Services 2 build.

This thread verifies the ability of the MDA service to:

- allow the viewing and updating of all of its schedule configuration parameters
- transfer the individual management data log files from each managed host, to the MSS server at each DAAC, on a scheduled basis
- retrieve the requested data and display it
- process the management data log files for the purposes of longer term analysis and reporting
- transfer log files to the ECS Data Archives

Test Configuration:

Hardware: MSS Server, Managed resources: host (i.e. SUN, HP) and applications, Inter-networking subsystem components (i.e. LAN Network resources)

Software: MCI

Data: Configuration data, management data log files

Tools: Xrunner, MDA user interface GUI

Test Input:

Inputs to this test case include inserting configuration data, processing time, archive time, keyins to remove entry from the configuration list, to update the configuration data, management data log files.

Test Output:

Outputs to this test case include displaying added new data, removes changes, changes, loading Counter Metrics, Duration Metrics, Summation/Average Metrics, Detail Events data to the Management RDBMS, retrieving and archiving requested data.

Success Criteria:

This test is successful when the MDA Graphical User Interface (GUI) allow to insert new configuration data, remove entry from the configuration list, update the configuration data. The MDA successfully processes the management data log files, loads these data to the Management RDBMS, retrieves requested data and transfers the log files to the ECS Data Archives.

Test Procedures:

Test Case ID: BC009.003	
Test Name: Management Data Access 2	
Test Steps:	Comments:
1. From the xterm, open the MDA user interface GUI.	MDA user interface GUI appears. The GUI has five buttons: MDA, Schedule&Configuration, Retrieve -Archive, Log Browser, and MDB Metrics.
2. On the MDA GUI, **select the Schedule&Configuration** button.	Two windows appear with Time, Task, Host and the other has CSS FTP, Process Log, Archive Log buttons.
3.**Select a Task** from the display	
4.**Select the CSS FTP Log**	The configDialogShell appears with the Task that has been selected. The following fileids appear: Host Name, Interval, Size Limit, Absolute Time, Absolute Time List, Selection.
5.**Change the Absolute Time value and update.**	The Absolute Time is updated with the new time.
6.**Click the Add button**.	The record is added to the Configuration.
7.**Select the same Task from the display.**	
8. **Select the CSS FTP Log**	The configDialogShell appears with the Task that has been selected. The following fileids appear: Host Name, Interval, Size Limit, Absolute Time, Absolute Time List, Selection.
9.**Click the Remove button.**	The record is removed from the Configuration.
10. **On the MDA GUI, select the Log Browser button.**	The Log Browser window appears.
11.**Click Select pulldown menu for processing.**	When Select pulldown menu is clicked, three cascade pulldown menus of single select filters (by Category, Priority, Subsystem,) is displayed with the Custom Select option.

12.**Click 'Category'**	When 'Category' is selected, five cascade pulldown submenus are displayed: Fault, Performance, Security Alert, Topology Change, and Transaction. When one of the options is selected, all events of the selected option is displayed in the events box.
13.**Click 'Priority'**	When 'Priority' is selected, four cascade pulldown submenus are displayed: Low, Medium Low, Medium, and High. When one of the options is selected, all events of the selected option is displayed in the events box.
14.**Click 'Subsystem'**	When 'Subsystem' is selected, eight cascade pulldown submenus are displayed: CSS, DMS, DPS, DSS, INS, IOS, MSS, and PDPS. When one of the options is selected, all events of the selected option is displayed in the events box.
15.**From Select pulldown menu, click 'Custom Select' or select the 'Custom Select' button at the bottom of the ECS Log Browser window.**	When the 'Custom Select' is clicked, the custom select window is displayed. (Currently Category, Priority, Subsystem, and Type are available for testing. but ApplicationID, CSCI, InstanceID, Mode, ProcessID, and ProgramID functionally will be provided in Phase 3.
16.**Click cancel to close window**.	
17. **Click Sort pulldown menu or Select pulldown menu for processing**.	If Sort pulldown menu is clicked, six choices of single sort and a Custom Sort for multi-level sorting will appear: Time Stamp, Category, Type, Priority, Subsystem, CSCI.
18.**Click 'Time Stamp'**	All events will be listed in chronological order. Note: This process will take a few minutes to generate due to the size of mss.log.
19.**Click 'Category'**	All events will be grouped in categories. Note: This process will take a few minutes to generate due to the size of mss.log.
20.**Click 'Type'**	All events with same types will be grouped and listed. Note: This process will take a few minutes to generate due to the size of mss.log.
21.**Click 'Priority'**	All events with the same priorities will be grouped and listed. Note: This process will take a few minutes to generate due to the size of mss.log.
22.**Click 'Subsystem'**	All events will be grouped by susbsystems and listed. Note: This process will take a few minutes to generate due to the size of mss.log.
23.**Click 'CSCI'**	All events will be grouped by CSCI and listed. Note: This process will take a few minutes to generate due to the size of mss.log.
24.**From the Sort pulldown menu, click 'Custom Sort' or click the 'Custom Sort' button** at the bottom of the ECS Log Browser window.	When the Custom Sort is selected, the Custom Sort window appears and it gives the user the option to select multi-level of sorting in forward and reverse order of the following: Time Stamp, Category, Type, Priority, Subsystem, and CSCI.
25.**If Edit pulldown menu is clicked,** two choices of undo and reset will appear.	Undo = go back once reset = go back to the original (1st list)
26. **Select the Retrieve - Archive button.**	The Retrieve - Archive dialog box appears with Host Name, a time option for Specific Time and Range Time, Start Time, Stop Time.

27.**Select Archive.**	The Archive window appears. C-MSS-18340
28.**Complete the following fields:** ***Host Name*** **(the machine you are on**) ***Specific Time*** ***Start Time*** ***Stop Time*** ***Click Submit***	The data is archived for the time period entered. Note the date and time ranges that is inputted.
29. **Select the Retrieve - Archive button**.	The Retrieve - Archive dialog box appears with Host Name, a time option for Specific Time and Range Time, Start Time, Stop Time.
30.**Select Retrieve**	The Retrieve window appears. C-MSS-18340
31.Complete the following fields: ***Host Name (*****the machine you are on*****)*** ***Specific Time*** ***Start Time*** ***Stop Time*** ***Click Submit***	The data is retrieved for the time period entered. Note the date and time ranges that is inputted.
32. Initialize the MDA user interface GUI to calculate the ***performance*** statistics.	The MDA user interface for performance is initialized.
33. ***Select*** **the MTBM (*Mean Time Between Maintenance*) button.**	MTBM selections appear.
34. ***Select and double click on one.***	The MTBM window opens and displays MTBM attributes.
35. Verify that the correct data is in the proper fields.	
36.***Save this file*** to verify that records can be appended to a log file.	The MTBM file is saved in the management database.
37. Open the saved file, it should be in a graph format.	File opens and a graph of MTBM is displayed.
38.***Select*** **the MTTR (*Mean Time to Repair*) button.**	MTTR selections appear.
39.***Select and double click on one.***	The MTTR window opens and displays MTTR attributes.
40.Verify that the correct data is in the proper fields.	
41. ***Save this file*** to verify that records can be appended to a log file.	The MTTR file is saved in the management database.
42.Open the saved file, it should be in a graph format.	File opens and a graph of MTTR is displayed.
43. ***Select*** **the MTBCM (*Mean Time Between Corrective Maintenance*) button.**	MTBCM selections appear.
44. ***Select and double click on one.***	The MTBCM window opens and displays MTBCM attributes.
45.Verify that the correct data is in the proper fields.	
46.***Save this file*** to verify that records can be appended to a log file.	The MTBCM file is saved in the management database.
47.Open the saved file, it should be in a graph format.	File opens and a graph of MTBCM is displayed.
48.***Select*** **the MTBPM (*Mean Time Between Preventive Maintenance*) button.**	MTBPM selections appear.
49.***Select and double click on one.***	The MTBPM window opens and displays MTBCM attributes.
50.Verify that the correct data is in the proper fields.	

51.***Save this file*** to verify that records can be appended to a log file.	The MTBPM file is saved in the management database.
52. Open the saved file, it should be in a graph format.	File opens and a graph of MTBPM is displayed.
53.***Select*** **the MDT (*Mean Down Time)* button.**	MDT selections appear.
54.***Select and double click on one.***	The MDT window opens and displays MDT attributes.
55.Verify that the correct data is in the proper fields.	
56.***Save this file*** to verify that records can be appended to a log file.	The MDT file is saved in the management database.
57.Open the saved file, it should be in a graph format.	File opens and a graph of MDT is displayed.
58. Initialize the MDA user interface GUI to calculate the ***fault*** statistics.	The MDA user interface for fault is initialized.
59. ***Select*** **the MTBM (*Mean Time Between Maintenance*) button.**	MTBM selections appear.
60. ***Select and double click on one.***	The MTBM window opens and displays MTBM attributes.
61.Verify that the correct data is in the proper fields.	
62.***Save this file*** to verify that records can be appended to a log file.	The MTBM file is saved in the management database.
63. Open the saved file, it should be in a graph format.	File opens and a graph of MTBM is displayed.
64.***Select*** **the MTTR (*Mean Time to Repair*) button.**	MTTR selections appear.
65.***Select and double click on one.***	The MTTR window opens and displays MTTR attributes.
66.Verify that the correct data is in the proper fields.	
67. ***Save this file*** to verify that records can be appended to a log file.	The MTTR file is saved in the management database.
68.Open the saved file, it should be in a graph format.	File opens and a graph of MTTR is displayed.
69.***Select*** **the MTBCM (*Mean Time Between Corrective Maintenance*) button.**	MTBCM selections appear.
70. ***Select and double click on one.***	The MTBCM window opens and displays MTBCM attributes.
71.Verify that the correct data is in the proper fields.	
72.***Save this file*** to verify that records can be appended to a log file.	The MTBCM file is saved in the management database.
73.Open the saved file, it should be in a graph format.	File opens and a graph of MTBCM is displayed.
74.***Select*** **the MTBPM (*Mean Time Between Preventive Maintenance*) button.**	MTBPM selections appear.
75.***Select and double click on one.***	The MTBPM window opens and displays MTBCM attributes.
76.Verify that the correct data is in the proper fields.	
77.***Save this file*** to verify that records can be appended to a log file.	The MTBPM file is saved in the management database.
78. Open the saved file, it should be in a graph format.	File opens and a graph of MTBPM is displayed.
79.***Select*** **the MDT (*Mean Down Time)* button.**	MDT selections appear.

80.***Select and double click on one.***	The MDT window opens and displays MDT attributes.
81.Verify that the correct data is in the proper fields.	
82.***Save this file*** to verify that records can be appended to a log file.	The MDT file is saved in the management database.
83.Open the saved file, it should be in a graph format.	File opens and a graph of MDT is displayed.
84. Exit the MDA user interface GUI.	The MDA user interface GUI window closes. DADS0901#A, PGS-0310#A, EOSD0510#A, EOSD0740#A, EOSD0750#A, EOSD0760#A, EOSD0500#A, EOSD1703#A, ESN-0830, ESN-1000, ESN-1070, ESN-0280, ESN-0290, ESN-0300, EOSD5110,

4.10.6.4 Test Case 4: Baseline Manager (BC009.004)

The Baseline Manager test is a regression test of the overall functionality of the Baseline Manager Thread. To ensure that the previously tested functionality of this thread remains intact upon integration with the Management Services 2 build.

This test verifies the ability of the Baseline Manager:

- to record and maintain configuration items including ECS deployed hardware and software and SCF-provided configuration data
- to maintain a version history of configuration controlled resources
- to forward records from local site to SMC
- to maintain records about each site and SMC configuration items
- at SMC to maintain information related to each sites baseline and distribute this information system-wide
- at SMC to maintain, and distribute to each site, baseline changes related to ECS resources deployed at each site
- application to be able to store accept information from a user interface, or from formatted data files
- application to be able to produce output data files containing information from user queries
- to maintain and log information including configuration management events and user specified events
- to integrate with Tivoli to perform the Resource Planner request

<u>Test Configuration:</u>

Hardware: LSM, SMC workstation

Software: Baseline Manager, Tivoli, Resource Planner

Data: LSM, SMC, ECS baseline information

Tools: N/A

Test Input:

Inputs to this test include application commands to enter, display, recall, view, and store baseline information, generate logs, and request resource configuration records.

Test Output:

Outputs include stored configuration/baseline records, version history of resources displayed to screen, transfer confirmation messages, configuration record lists, configuration items reports, user requested reports, site specific reports, baseline information, confirmation of data ingest, event logs, and a popup window on the planning workstation that includes the resource configuration record information.

Success Criteria:

This test is successful when the configuration information and site's operation baseline are retrievable. The user is able to recall version histories of the baseline resources and receive site specific reports. Baseline information viewed matches any changes that were recorded from the SMC baseline manager. The baseline manager accepts data entry and stores all information, and produces data files which contain user requested baseline information. Logs generated include specified CM and user specified events. Also, the resource configuration record information in the popup window corresponds to the user query, and the interaction between the Resource Planner, Tivoli, and the Baseline Manager work properly.

Test Procedures:

Test Case ID: BC009.004	
Test Name: Baseline Manager	
Test Steps:	Comments:
a. Maintenance of Configuration Items	
1. Log into each LSM and SMC workstation and execute Baseline Manager application. **Xrp**	Baseline Manager application will appear.
2. Execute commands to enter ECS baseline information.	Baseline information includes hardware, software, documentation, and science algorithm items.

3. Verify the baseline manager tracks the following: a. ECS subsystems, networks, and configured system and network devices such as workstations, servers, and routers. b. ECS releases and site baselines c. ECS hardware and software resources designated as configuration items. d. specifications associated with configuration items e. technical documentation and test materials f. scientific algorithms, including software, data and test materials	SMC-2510 (partial), SMC-2515 (partial), SMC-6340 (partial)
4. Verify the baseline manager tracks and maintains SCF-provided configuration data including: a. algorithm development version numbers, identification codes and reference numbers b. SCF point of contact's name and organization. c. associated files' names, formats, sized, and descriptions d. number of files by category and type.	SMC-2510 (partial), SMC-2515 (partial) PGS-0950
5. Verify the baseline manager tracks and maintains software-critical and security sensitive items.	SMC-2510 (partial)
5. Submit information into baseline manager to store and update.	Stores all baseline information into database.
b. Forwarding Baseline Records to SMC	
6. On the LSM workstation identify and access all site operational baseline and the version and status of configuration controlled resources related to the baseline. Submit this information for transfer to the SMC.	SMC-2515 (partial)
7. On the LSM workstation identify a "level of assembly" list that describes the composition of configuration items and submit this information for transfer to the SMC.	SMC-2515 (partial)
8. Transfer all information to the SMC.	Records and lists specific to the LSM will be transferred to the SMC baseline manager.
9. From the SMC baseline manager, verify all information from LSM baseline manager was transferred and stored to SMC database.	Verification that records and lists from the LSM were transferred and stored to the SMC baseline manager. All information will be analyzed for completeness.
10. From the SMC baseline manager, verify that ECS resources are characterized as system-wide or site-wide.	DADS0901#A, PGS-0310#A, EOSD0510#A, EOSD0740#A, EOSD0750#A, EOSD0760#A, EOSD0500#A, EOSD1703#A
11. Exit baseline manager.	

4.10.6.5 Test Case 5: Physical Configuration Management (BC009.005)

The Physical Configuration Management test is a regression test of the overall functionality of the Physical Configuration Management Thread. To ensure that the previously tested functionality of this thread remains intact upon integration with the Management Services 2 build.

This test will verify the ability to:

- view the physical location of system components on the floor plans and changing the location of the system components using the PNM Physical CM COTS tool
- generate standard and customizable reports of Physical CM data using the PNM Physical CM COTS tool
- import and store an existing system floor plan
- add a new node to the system, move an existing node to a new location in the system, check the nodal attributes in the database and delete an existing node from the system

Test Configuration:

Hardware: MSS Server

Software: Physical Network Manager/Mountain View (Physical CM COTS package), Sybase

Data: Hardware and Software Resource Objects

Tools: HpOpenView (Physical location of resources)

Test Input:

Operator actions to exercise the capabilities of the GUI, to generate Physical CM reports, to enter data item information into the CM tool, and to add a node to the network.

Test Output:

Displays, printouts, physical CM reports and notification that a node has been added to the network.

Success Criteria:

This test is successful if GUI-specific actions and pull-down screens correctly respond to user input, Physical CM reports are created and contained proper information and a new node is added to the network. Nodal attributes are automatically gathered and added to the database, additional information can be added manually, and the physical map is updated to include the new node symbol. The information concerning the new location of the node is configured into the system. Deletion of a node from the system and that the node symbol on the network map is removed with all its database links deleted.

Test Procedures:

Test Case ID: BC009.005	
Test Name: Physical Configuration Management	
Test Steps: Maintain Physical CM Records	Comments:
	CONVENTIONS USED IN THIS PROCEDURE: Mouse button keys are indicated as follows: LMB - Left mouse button MMB - Middle mouse button RMB - Right mouse button
a. Import an existing floor plan.	
1. Click RMB to bring up the Mtp Main Menu.	The Mtp-Main Menu window will appear.
2. Select **TOOLS>SCAN>FILE>LOAD IMAGE**	The systems displays the default directory path in the input window and its files or subdirectories in the menu.
3. Enter **/home/<username>/pnm/newdraw** in window box click *<filename>.*	Imported scanned image drawing will appear in small scale.
4. Select **EXIT>DISPLAY>FULL VIEW**	Full scale drawing will be displayed.
5. Click MMB to remove the Mtp-Main Menu box to view the image.	
6. Click RMB to bring up Mtp-Main Menu box and Select **TOOLS>SCAN>EDIT> EXIT DRAW>DELETE>DELETE ALL**	The Mtp-Main Menu window will appear. To exit the drawing.
b. Add physical components to imported floor plan.	
7. Select from Top-Menu **Designer**, from Sub-Menu select **Add Object>Devices**	Allows you in the network mode for interfacing with network objects.
8. Select in box <pick workstation>.	Window will appear for devices selection.
9. Select **Accept** in window box.	Updates status in the database.
10. Resize object press **"?/"** key until object reaches desired size and click LMB to place drawing.	Object is place on the floor plan.
11. Select from Top-Menu **Save File**.	
12. Select from Mtp-Main Menu **Save and Exit**.	The Mtp-Main Menu appears.
13. Enter **/home/<username>/pnm/newdraw/ <***filename***>**	The Mtp-Main Menu window will appear.
14. Select from Top-Menu **Load File**, enter your **<***filename***>.**	Load Mtp Menu box will appear.
15. Select Attach button and select the object and yes with LMB.	
16. Select from Top-Menu **Designer**, from the Sub-Menu select **View Object**	Allows you to view information about the selected component
17. Select object and enter information.	
c. Store the component information	
18. Select **Accept** twice in window boxes.	Updates status in the database.
19. Select from Top-Menu **Network Manager**, from the Sub-Menu select **Update Status>One Node**	Allows you to update the network status for a selected single node.
d. Load information for ECS components to reflect inventory, network and maintenance	
20. Click RMB to bring up the Mtp-Main Menu.	The Mtp-Main Menu window will appear.
21. Select **TOOLS>RISE>DB ADD>ADD ENTRY RECORD**	Allows you to add operations pertaining to components.

22. Select object and click **YES** with LMB.	The Select table window will appear.
23. Select **MTPH1** table and enter **<***information***>.**	
24. Exit the table on softkeys **Exit Editor**.	Entry window box appear.
25. Select **YES** in window Are you sure you want to add this entry?	
e. Search and retrieve information from the tables via PNM SQL search language.	
26. Click RMB to bring up the Mtp-Main Menu.	The Mtp-Main Menu window will appear.
27. Select **TOOLS>RISE>DB SEARCH> EZY QUERY**	The Database table window will appear.
28. Select the MTPH1 table and click LMB.	Brings up SQL database information.
29. Select boxes that you want to retrieve information.	
30. Select Continue 3 times (lower right hand corner).	The new updated information is in database.
31. Click MMB to exit out of table.	
32. Click RMB to bring up the Mtp-Main Menu.	The Mtp-Main Menu window will appear.
33. Select **TOOLS>SCAN>EDIT>DELETE IMAGE** Do you want to Delete Image Select **NO**.	Remove you from the image window.
f. Generate a standard set of reports against this data	
34. Select **Load File** from Top-Menu option	Load Mtp Menu box will appear.
35.Enter **/home/<username>/pnm/newdraw/<Daac>** in the input line window <RET>.	The selected drawing will appear.
36. Click RMB to bring up the Mtp-Main Menu.	The Mtp-Main Menu window will appear.
37. Select **TOOLS>RISE>DB ADD> MODE:FORM>ADD ENTRY RECORD**	Mode change allows the user to switch from "Normal" to "Form" mode for report generation.
38. Click MMB to close Mtp-Main Menu window.	
39. Select the desire object and say yes.	The Database table window will appear.
40.Select table (report) click LMB.	
41.Select on softkeys **EXIT Editor** from table.	Withdraws you from the table.
g. Generate a custom set of reports against this data	C-MSS-87610
42. Select **Load File** from Top-Menu option	Load Mtp Menu box will appear.
43.Enter **/home/<username>/pnm/newdraw/<Daac>** in the input line window <RET>.	The selected drawing will appear.
44. Click RMB to bring up the Mtp-Main Menu.	The Mtp-Main Menu window will appear.
45. Select **TOOLS>RISE>DB ADD> MODE:FORM>ADD ENTRY RECORD**	Mode change allows to switch from Normal to Form mode for report generation.
46. Click MMB to close Mtp-Main Menu window.	
47. Select the desire object and say yes.	The Database table window will appear.
48.Click LMB and the status information on object appears.	Window box appears to add an entry to to table (report).
49. Are you sure you wish to add this entry? Select **YES**	Allows you editing information associated with the report environment.
50. Enter in item <*information*>.	

51. Select on softkeys **EXIT Editor** from table.	Exits table. DADS0901#A, PGS-0310#A,EOSD0510#A, EOSD0740#A, EOSD0750#A, EOSD0760#A, EOSD0500#A, EOSD1703#A, SMC-2305, SMC-2315, SMC-2325, SMC-2335, SMC-2515, SMC-2535, SMC-6345,

4.10.6.6 Test Case 6: Trouble Ticketing (BC009.006)

The Trouble Ticketing test is a regression test of the overall functionality of the Trouble Ticketing Thread. To ensure that the previously tested functionality of this thread remains intact upon integration with the Management Services 2 build.

This test will verify the ability to:

- notify the originator when the status of the submitted trouble ticket has been modified
- search by an internal user for historical and current trouble tickets using keywords, user id, and trouble ticket id
- forward by the internal user a selected trouble ticket from one organization to another
- maintain different trouble ticket status's to include Open, Work-In-Process, Closed, and Archived
- search by the internal user for trouble tickets relating to the same source
- store information that is pertinent to trouble tickets, and retrieve this information at a later time
- generate reports from its database and print to a screen or printer
- provide customization features that enable internal users from other locations to specify notification and escalation rules
- support the entry, submission, and editing of trouble tickets by a registered user through graphical user interface
- enter trouble tickets from another application program

Test Configuration:

Hardware: HP Server

Software: Management Services (Remedy Package)

Data: Sample problem to report, including a short problem description and severity number

Tools: XRunner/LoadRunner

Test Input:

Inputs include the creation of trouble tickets that are modified throughout the course of the test, and a simulated SMC E-Mail address.

Test Output:

Notification via E-Mail that the Trouble Ticket's status was created and has been modified.

Success Criteria:

The test is successful when trouble tickets are created, modified, notified via E-Mail that the Trouble Ticket's status has been modified and selected trouble tickets are forwarded to another organization and receipt of delivery has been returned.

Test Procedures:

Test Case ID: BC009.006	
Test Name: Trouble Ticketing	
Test Steps:	Comments:
a. Notification of Status Changes	
1. Log onto ECS Trouble Ticket Service.	
2. Select submit a trouble ticket.	
3. Verify submit a trouble ticket homepage appears.	
4. Enter the User Information.	
5. Enter the Problem Information.	
6. Select submit icon.	
7. Return to ECS Trouble Ticket Service Home page.	
8. Log-off as user.	
9. Log-on as Internal user.	
10. Select Release-A Action Request System.	
11. Enter the ticket-id.	
12. Select query-list.	
13. Select query-modify individual.	
14. Verify that the Modify individual homepage appears.	
15. Modify the ticket status to closed.	
16. Select closing code.	
17. Select apply.	
18. Select Dismiss.	
19. Log-off as Internal user.	
20. Log-on as user.	
21. Verify that an E-Mail notification message has been received.	
b. Forwarding Trouble Tickets	
22. Log onto ECS Trouble Ticket Service as Internal user.	
23. Enter ticket-id.	
24. Verify that the selected ticket-id was retrieved.	
25. Forward the trouble ticket to a technician, help-desk, or another site.	

26. Enter data into forward-to, forward-from, forward-by, and forward-date cells.	
27. Log-off as internal user.	
28. Log-on as technician.	
29. Verify that the selected trouble ticket was forwarded.	
c. Trouble Ticket Report	
30. Select Query-list to display a list of Trouble Tickets	
31. Select a Trouble Ticket	
32. Query report of Trouble Ticket selected	
33. Verify report was generated	
34. Select print to screen or printer	
35. Verify report was printed to screen or printer	DADS0901#A, PGS-0310#A, EOSD0510#A, EOSD0740#A, EOSD0750#A, EOSD0760#A, EOSD0500#A, EOSD1703#A
36. Log-off	

4.10.6.7 Test Case 7: Software Change Manager (BC009.007)

The Software Change Manager test is a regression test of the overall functionality of the Software CM Thread. To ensure that the previously tested functionality of this thread remains intact upon integration with the Management Services 2 build.

This test verifies the ability to:

- maintain various types of software library files by using Clearcase checkout/edit/checkin paradigm
- implement properly ECS policies as it relates to transitioning configured file elements through software lifecycle promotion phases
- create a build, label the components of the build, create a branch, modify a file element, merge the edited library file with the existing one and identified version conflicts, if necessary
- allow users to work concurrently on the same library file element and identify version conflicts

Test Configuration:

Hardware: SUN clearcase client and server, HP client, SGI client

Software: Clearcase (COTS), Custom Trigger scripts, Custom software library policy scripts, authorization tables

Data: Sample Site Software library items, Sample Site Software library file elements to represent science algorithms, Sample Site Software library file elements that comprise a build

Tools: N/A

Test Input:

Inputs to this test case include a representative sampling of software library files, clearcase commands to invoke scripts, to manipulate configured file elements, to operate on a specific library file element, and file elements to comprise a build.

Test Output:

Outputs of this test case consist of ClearCase messages, version control numbers, attributes, labels appended to file elements, resulting logs, status messages, file elements with state and promotion level attributes, custom script log messages, resulting ClearCase logs.

Success Criteria:

This test case is successful when a sample of the stipulated test input is checked in to the Software Change Manager. Attempts to change state and promotion level attributes of file elements are allowed when the user has been granted the correct permissions. Attempts to modify attributes of software library elements is denied when authorization has not been granted. A build is successfully performed and two users are able to modify library file elements concurrently.

Test Procedures:

Test Case ID: BC009.007	
Test Name: Software Change Manager	
Test Steps:	Comments:
a. Verify Maintenance of Software Library Elements	Note: throughout this procedure "c" is an alias for the "cleartool" command
1. Log in to one of the supported architecture machines. Enter : *<login id>* on the workstation Enter: *<password>* on the workstation	This test procedure is to be performed on 3 different hardware architectures: SGI HP SUN
2. Set clearcase view and show config spec **c setview** *<username>* **c catcs**	 Displays the configuration spec used with the "username" view
3. Checkout the configured directory **cd /ecs/SCMscience/<platform>/** **c co -nc .**	Configured directory represents a software library. C-MSS-40480

4. Copy file elements to a configured directory **cp /home/nwoodson/SCM/testdata/ELEMENTS/* .**	Copies data elements: DbDeleteDriver.input DbDeleteDriver.output DpAtEnv.csh.template DpAtExecution.cxx DpAtMMgrXdiff.sh DpAtPcFileInfo.cxx DpPrExecUnitTest.plan DpPrExecutable.doc spre4sun.format.partition spre4sun.infoscript to the ClearCase configured directory
5. Verify that the files exist but are not recognized as "Clearcase" elements **ls** **c ls**	Perform a unix list to verify that files do exist Perform a cleartool list to show that Clearcase does not acknowledge these as elements known to the configured directory
6. Make the data elements "ClearCase" elements. **c mkelem -ci -nc ***	Use the cleartool makelement command to make all the files in the configured directory "ClearCase" elements and check them back in.
7. Verify that the files in the directory are now recognized as "Clearcase" elements. **c ls:**	Perform a cleartool list to show that ClearCase now acknowledges that file elements as "ClearCase" elements
8. Verify by inspection that the files all have been assigned version numbers by viewing the output to the screen resulting from the "c ls" command..	
9. Checkout the configured directory **c co**	
10. Make the files elements in the configured directory Clearcase elements **c mkelem -ci -nc ***	triggers on the mkelem command create state, promotion level and checksum attribute*s*.
11. Verify by inspection the version numbers of configured items in the software library **c ls** **c lshistory -long**	produces log of software library files and their version numbers.
12. Check the configured directory back in **c ci -nc** .	
b. Verify Software Build Procedures	
13. Go to the Clearcase VOB directory **cd /ecs/SCMscience**	
14. Go to directory of the host platform **cd** <*platform*>	ie: cd SUN5 cd HP cd SGI
15. Check out the platform directory **c co -nc** .	Checks out the platform directory so that a BUILD directory can be created below it.

16. Make a clearcase directory named “BUILD” to keep build elements **c mkdir -nc BUILD**	Creates a ClearCase configured directory.
17. Check in the platform directory **c ci -nc** .	Checking in the platform directory allows pubic to see platform directory file elements.
18. Change your cwd to BUILD **cd BUILD**	
19. Copy all build elements to the BUILD directory **cp /home/nwoodson/SCM/testdata/ELEMENTS/*time* .**	Copies : make.time make file time3.cxx source file time3.h include file time3drv.cxx driver build elements to the ClearCase configured directory.
20. Make all build files clearcase elements and check them in **c mkelem -ci -nc ***	Triggers on the mkelem command create state, promotion level and checksum attributes
21. Verify by inspection that the build elements in the build directory are all “ClearCase” elements. **c ls**.	
22. Check in the BUILD directory **c -ci -nc**	
23. Verify by inspection that the BUILD directory is a “ClearCase” element. **c ls**.	
24. Move to <*platform*> directory **cd ..**	Move to the ecs/SCMscience/<*platform*> directory so that we can later change our config spec.
25. Make a branch **c lstype -brtype** **c mkbrtype -c “platform dev branch” <platform>_build**	Lists all the existing branches made by the user or any other user. For example: Creates the branch “sun_build” and attaches the title “platform dev branch” to it.
26. Reset your view and display **c setcs /home/username/SCM/<platform>_view** **>c catcs**	Resets the config spec to element * CHECKEDOUT element * /main_<platform>/LATEST element * /main/LATEST -mkbranch <platform>_build Displays the config spec
27. Move to the BUILD directory and list the clearcase elements in that directory **cd BUILD** **c ls**	
28. Check out all the build elements needed onto the new branch created in step 16 and list **c co -nc** * **c unco *** **c ls**	Checks the build elements out on the <platform>_build branch
29. Build the executable without configuration records using a make file **clearmake -C gnu -F -f make.time**	

30. Move the executable to another file and delete object files. **mv timedrv timedrv**_<platform>.**tc1.nocr** **rm *.o**	Object files are deleted so that the next build will rec Removes object files compile.
31. Rebuild the executable with configuration records **clearmake -C gnu -f make.time** **mv timedrv timedrv**_<platform>.**tc1**	
32. Display the history and audit trail of the build elements and store it in a file **c catcr -long -r timedrv**_<platform>.**tc1 <platform>_build1.audit**	
33. Attach the label "BUILD1" to all build elements **c mklabel -config timedrv**_<platform>.**tc1 BUILD1**	
34. Verify that the "BUILD1" label was applied to all file elements in the build **c describe * >** <platform>**_build1.label**	
35. Do a unix list and a ClearCase list **ls** **c ls**	
36. Display current config spec **c catcs**	
37. Change the config spec **c setcs /home/username/SCM/<platform>_bug_view**	Change the config spec to enable the editing of a file in the build as a bug fix and merge it back to the <platform>_build.
38. Display the new config spec **c catcs** **c ls**	The config spec should be changed to: element * CHECKEDOUT element * /main/<platform>_build/bug_fix/LATEST element * /main/<platform>_build/LATEST -mkbranch bug_fix element * /main/LATEST -mkbranch sun_build
39. Check out a file element **c co -nc time3.cxx**	Checks out time3.cxx on the bug_fix branch
40. Verify by inspection that the file element is checked out on the bug_fix branch **c ls**	
41. Copy an edited version of a file element (in the build) over the existing copy of the file element **cp /home/username/tme3.cxx.temp time3.cxx**	Simulates editing a version of a file element
42. Check the new file back in to ClearCase **c ci -nc time3.cxx** **c ls**	Creates a new version of the time3.cxx file and assigns version numbers
43. Build the executable and do not create configuration records. **clearmake -C gnu -F -f make.time** **mv timedrv timedrv**_*<platform>*.**tc2.nocr**	
44. Remove all object files **rm *.o**	

45. Rebuild executable and create configuration records **clearmake -C gnu -f make.time** **mv timedrv timedrv**_<*platform*>**.tc2**	
46. Display the history and audit trail of the build elements and store it in a file **c catcr -long -r timedrv**_<*platform*>**.tc2** <*platform*>_**build2.audit**	
47. Return to the previous view **c setcs /home/username/SCM**/<platform>_**view** **c catcs** **c ls**	 Display the new view Display all the "ClearCase:" elements
48. Merge executable from the bug_fix branch to the <platform>_build branch **c co -nc time3.cxx** **c merge -to time3.cxx -version /main/<***platform*>_**build/bug_fix/LATEST** **c ls**	 Check out time3.cxx to merge to the <platform>_build branch Merges time3.cxx from bug_fix branch to <platform>_build branch
49. Check the merged version of the executable back in **c ci -nc time3.cxx** **c ls**	
50. Verify by inspection that file labels, file history and merge history are maintained on the modified build element. **c lshistory time3.cxx**	
51. Build the executable once again without config records based on the merged version of the executable **clearmake -C gnu -F -f make.time** **mv timedrv timedrv**_<*platform*>.**tc3.nocr** **rm *.o**	
52. Rebuild the executable and generate config records based on the merged version of the executable **clearmake -C gnu -f make.time** **mv timedrv timedrv**_<*platform*>.**tc3**	
53. Display the history and audit trail of the build elements and store it in a file **c catcr -long -r timedrv**_<*platform*>**.tc3** <*platform*>_**build3.audit** **c ls**	DADS0901#A, PGS-0310#A, EOSD0510#A, EOSD0740#A, EOSD0750#A, EOSD0760#A, EOSD0500#A, EOSD1703#A

4.10.6.8 Test Case 8: System Monitoring (BC009.008)

This test verifies the system monitoring functionality of external and internal data transfers, user sessions, network configurations, policies and procedures, and events. The system monitors internal and external data transfers between system components and errors are induced into the data transfer process to verify error detection. Log files are updated to reflect the successful transfers and the failed attempts.

This test also verifies that user sessions are monitored and that HpOpenView monitors and displays the local configuration status. Policies and procedures within the ECS system are viewed and modified, and each element schedule and event is monitored and the display updated as events occur and are added to the schedule.

Also verified is the ability for ECS elements to be capable of verifying the fidelity of the ECS element interface to:

- Other ECS elements at any time during the lifetime of the ECS
- Entities external to ECS at any time during the lifetime of the ECS

Test Configuration:

Hardware: MSS-EDF, DCE Configured LAN

Software: MLCI, MCI, MACI

Data: Sample data for transfer

Tools: Monitoring and logging tool, HpOpenView

Test Input:

Inputs include data transfer commands and interruption, user session and network configuration requests, modification of policy and procedure display and schedules and events.

Test Output:

Log file updates, alarms, relinquished connections, display of local and system network configuration, and displays of policy and procedures.

Success Criteria:

This test is successful if the system monitors the transfer of internal and external data and HpOpenView detects any failed transfers. Also required is the system monitoring of user sessions and expiration of inactive sessions after a predetermined time. HpOpenView must reflect the current system configuration, and the policy and procedures are viewed and altered as applicable. Also, as events occur, the displays are updated to reflect event completion and new events added to the schedule are reflected on the displays. The log file must be updated to reflect the transferred data and the failed transfer requests, and the expired sessions.

Test Procedures:

Test Case ID: BC009.008	
Test Name: System Monitoring	
Test Steps:	Comments:
1. Transfer data with induced errors between system components.	Expect log files are updated to reflect the successful transfers, the failed attempts, and the expired sessions.
2. Interrupt the data transfer.	Expect alarms and relinquished connections.
3. Request user session and network configuration.	Verify HpOpenView monitors and displays the local and system network configuration status.
4. Modify policy and procedures.	Expect HpOpenView displays policy and procedures. When events occur, the displays are updated to reflect new events added to the schedule.
	DADS0901#A, PGS-0310#A, EOSD0510#A, EOSD0740#A, EOSD0750#A, EOSD0760#A, EOSD0500#A, EOSD1703#A, SMC-3305, EOSD0780, EOSD4100, ESN-0010, ESN-0800, ESN-1090, ESN-1140, ESN-0620, ESN-0650

4.10.6.9 Test Case 9: LSM Scenario (BC009.009)

The purpose of the LSM Scenario test is to verify the LSM. The testing will take place at the EDF in an environment that emulates the planned production LSMs. DAAC and SMC functionality required to support LSM testing will also be emulated in the EDF.

Test Configuration:

Hardware: MSS-EDF

Software: MACI, MLCI, MCI

Data: simulated data

Tools: X-Runner, Load-Runner

Test Input:

Inputs to this test case include the insertion of faults, simulated data on the operational states of applications, operating system resources and network components, analyzed data compared with established criteria, and adjusted measurement criteria. Also input is the insertion of a new security rule/policy, security intrusions, password audits, and CCRs.

Test Output:

Electronic copies of changes required by the CCR, summary reports about change requests associated with each site, indication (either screen notification or E-mail) that a fault has occurred, logs containing historical and statistical data, local performance data periodically sent to the SMC, message sent out to other DAACs and SMC notifying them of intrusion attempts and reports listing security events and intrusion notification.

Success Criteria:

This test is successful when the following events can be performed without error:

- A CCR is entered into the system and the proper steps/procedures are followed throughout closure. This results in new software checked into the CM tool and configured properly
- A fault occurs and the fault is recognized and the necessary recovery steps are taken
- Appropriate performance data is generated locally and periodically sent to the SMC
- Intrusion attempts are logged and the SMC is appropriately notified of the attempts

Test Procedures:

Test Case ID: BC009.009	
Test Name: LSM Scenario	
Test Steps:	Comments:
1. Inject a fault at the emulated LSM located in the EDF. A variety of faults will be incurred in a variety of ways.	Verify that the LSM receives appropriate notification of the fault condition from the Fault Management Service. Verify the SMC receives notification of the fault sent by LSM.
2. Produce reports based on the information collected from the various Fault Management Application Services.	Verify that the fault is recognized and the necessary recovery steps are taken.
3. Verify that the site Performance Management Application Service collects and processes performance data local to the LSM and periodically sent to the SMC.	
4. Produce the LSM summary performance reports and notifications of performance metrics exceeding established thresholds.	
5. Log into an emulated LSM (at the EDF) not meeting all of the criteria defined in the security directives and guidelines by the SMC.	Verify that the user gets flagged and that the information is recorded at the LSM
6. Intrusion attempts and incorrect password entered.	The attempts are logged and the SMC is appropriately notified of the attempts.
7. Verify that the recovery from detected security breaches at the sites.	Upon recovery, verify that the LSM may coordinate recovery from security events via electronic mail and the telephone.
8. Verify that the User Registration and Accountability Audit Trail send reports to the sites.	
9. Check a new file into the CM tool.	
10. Verify that the Software Change Manager automatically creates and safeguards a file version whenever a new or changed file is checked into the library.	
11. Open a new CCR.	Verify the proper steps/procedures are followed throughout closure.

	DADS0901#A, PGS-0310#A, EOSD0510#A, EOSD0740#A, EOSD0750#A, EOSD0760#A, EOSD0500#A, EOSD1703#A, SMC-2505, SMC-2515, SMC-2535, SMC-2605, SMC-3305, SMC-3335, SMC-3345, SMC-3355, SMC-3375, SMC-3385, SMC-3395, SMC-3397, SMC-3415, SMC-4305, SMC-4315, SMC-4325, SMC-4335, SMC-5305, SMC-5325, SMC-5335, SMC-5345, SMC-5355, SMC-5365, SMC-6315, SMC-6325, SMC-6335, SMC-6345, SMC-8305, SMC-8705,

4.10.6.10 Test Case 10: General DBMS (BC009.010)

The purpose of the General DBMS test case is to demonstrate/inspect all of the available functionality associated with the Release A DBMS. This test also verifies that the Monitor/Control Service successfully performs statistical analysis and logs resultant information into the database.

Test Configuration:

Hardware: LAN Analyzer, DCE Configured LAN

Software: Protocol Decoder, Network Management, DCE Software/Security Server

Data: Simulated data

Tools: Monitoring and logging tool, HpOpenView, XRunner/LoadRunner

Test Input:

Inputs to this test case include commands to initialize various DBMS functionality and data to perform statistical analysis. Specific steps include keyins to accomplish the following:

- Initialize the DBMS
- Verify that the DBMS requires security access control based upon user id, role and privileges for database, database objects, and database operations
- Demonstrate that the DBMS provides an SQL interface with query, update, and administrative functions
- Inspect the Federal Information Processing System Publication to ensure compliance with SQL-2
- Initialize the related CSS session-establishment service, verify that the DBMS can be accessed by or supported by the service
- Verify that a client/server paradigm is supported
- Generate ad hoc statistics from the management data

- Demonstrate the functionality for bulk data load and database backups (frequency, time, and type of backups)
- Demonstrate on-line disk management functionality
- Verify the compatibility of the DBMS and the ECS management framework to support the importing of the ECS management framework data
- Verify that access structures are supported to improve the efficiency of retrieval of management data
- Verify data compression, space reallocated from deleted records, and variable-length column storage
- Restore all or part of the database to a specific time
- Demonstrate the functionality associated with recovery
- While recording data, issue and record a database checkpoint
- Run an audit trail to verify that all of the database activities have been recorded
- Perform statistical analysis by the Monitor/Control Service
- Verify that the statistical analysis performed by the Monitor/Control Service can be stored in the management database for historical purposes
- Verify that the Statistical analysis is for average, median, maximum, minimum, ratios, rates and standard deviation

Test Output:

The outputs to this test case include ad hoc statistics and reports, results of database queries, updated databases, database backups, archived data, recovered data, data compression and bulk loaded databases, and an audit trail reflecting database operations. Database output includes statistical analysis results stored in the management database.

Success Criteria:

This test is successful when all of the functionality associated with the data base has been demonstrated, tested, inspected, and verified. This includes statistical analysis results and logging of data into the management database.

Test Procedures:

Test Case ID: BC009.010	
Test Name: General DBMS	
Test Steps:	Comments:
1. Initialize the DBMS.	

2. Verify that the DBMS requires security access control.	
3. Demonstrate that the DBMS provides an SQL interface with query, update, and administrative functions.	
4. Initialize the related CSS session-establishment service, verify that the DBMS can be accessed by or supported by the service.	
5. Verify that a client/server paradigm is supported.	
6. Generate ad hoc statistics from the management data.	
7. Demonstrate the functionality for bulk data load and database backups (frequency, time, and type of backups).	
8. Demonstrate on-line disk management functionality.	
9. Verify the compatibility of the DBMS and the ECS management framework to support the importing of the ECS management framework data.	
10. Verify that access structures are supported to improve the efficiency of retrieval of management data.	
11. Verify data compression, space reallocated from deleted records, and variable-length column storage.	
12. Restore all or part of the database to a specific time.	
13. Demonstrate the functionality associated with recovery.	
14. While recording data, issue and record a database checkpoint.	
15. Run an audit trail to verify that all of the database activities have been recorded.	
16. Perform statistical analysis by the Monitor/Control Service.	
17. Verify that the statistical analysis performed by the Monitor/Control Service can be stored in the management database for historical purposes.	
18. Verify that the Statistical analysis is for average, median, maximum, minimum, ratios, rates and standard deviation.	
	DADS0901#A, PGS-0310#A, EOSD0510#A, EOSD0740#A, EOSD0750#A, EOSD0760#A, EOSD0500#A, EOSD1703#A, SMC-2505, SMC-6325, EOSD2440, EOSD3200, EOSD5020, EOSD5040, EOSD5200, EOSD5250, ESN-0003 C-MSS-90020, 90060, 90070, 90160, 90180, 90170, 90120, 90030, 90080, 90290, 90280, 90260, 90240, 90230, 90210, 90200, 90190, 90150, 90140

4.10.6.11 Test Case 11: Report Generator (BC009.011)

The purpose of the Report Generator test case is to demonstrate/inspect all of the available functionality associated with the report generator.

Test Configuration:

Hardware: LAN Analyzer, DCE Configured LAN

Software: Protocol Decoder, Network Management, DCE Software/Security Server

Data: Management data

Tools: Monitoring and logging tool, HpOpenView, XRunner/LoadRunner

Test Input:

Inputs to this test case include but are not limited to: all available reports, inputs to create ad hoc reports, management data maintained in the DBMS and the ability to redirect reports to a different device. Operator actions will be conducted to perform the following:

- Verify that the report generator can receive information from the DBMS and generate reports from this information
- Demonstrate the existence of a Motif based Graphical User Interface (GUI) and the functionality associated with using the GUI to generate reports
- Demonstrate the ability of the report generator to generate ad hoc reports from the managed data maintained in the DBMS
- Verify that the reports generated can be formatted to contain any/all of the following information: title, header, footer, page number, date/time of report
- Demonstrate the capability to create charts and graphs from the management data contained in the DBMS
- Verify the ability of the report generator to redirect the output to the console, a disk file or printer

Test Output:

Outputs to this test case include a variety of reports and charts directed to various output devices.

Success Criteria:

This test is successful when the GUI is successfully used to generate a variety of reports based on the management data and these reports are capable of being output to a variety of different output devices.

Test Procedures:

Test Case ID: BC009.011	
Test Name: Report Generator	
Test Steps:	Comments:
1. Verify the report generator can receive information from the DBMS and generate reports from this information.	
2. Demonstrate the existence of a Motif based Graphical User Interface (GUI) and the functionality associated with using the GUI to generate reports.	
3. Demonstrate the ability of the report generator to generate ad hoc reports from the managed data maintained in the DBMS.	
4. Verify that the reports generated can be formatted to contain any/all of the following information: title, header, footer, page number, date/time of report.	
5. Demonstrate the capability to create charts and graphs from the management data contained in the DBMS.	
6. Verify the ability of the report generator to redirect the output to the console, a disk file or printer.	
	DADS0901#A, PGS-0310#A, EOSD0510#A, EOSD0740#A, EOSD0750#A, EOSD0760#A, EOSD0500#A, EOSD1703#A, SMC-2115#A, SMC-2415#A, SMC-8305#A, SMC-8705#A, ESN-0770#A, C-MSS-90520, 90530, 90500, 90570, 90600, 91010, 90510

4.11 Communications Infrastructure Three Tests

The Communications Infrastructure 3 tests are a collection of tests designed to verify the integration of the Communication Infrastructure 2 Build with the following threads and builds:

- Electronic Mail (E-mail) /Bulletin Board Service (BBS) Thread (TC032)
- Virtual and X-Terminal Support Thread (TC033)
- Managed FTP Thread (TC034)
- Communications Infrastructure 3 Build (BC010)

4.11.1 Electronic Mail (E-Mail) /BBS Thread (TC032)

The purpose of this thread is to verify the software specifications of the Electronic Mail (E-mail) Service and the Bulletin Board Service (BBS). The E-mail Service implements the CsEmMailRelA class to send electronic messages via the sendmail system command between host(s). The BBS implements the CsBBMailRelA class to post messages and files on destination groups utilizing a COTS package called the NN.

The E-mail Service provides the capability to manage electronic mail messages to its users and applications. The service supports the Simple Mail Transport Protocol (SMTP) and uses the existing X.400 Open Systems Interconnection (OSI) standard/SMTP gateway at Goddard Space Flight Center (GSFC).

The E-mail Service implementation includes a mail server to respond to incoming, outgoing, and forwarded messages; mail manager to provide total maintenance of mail system; message body to edit and view messages; attachments to view, edit, and discard appendices; mailing list for destination modifications; folder as archive for message summaries and actions; mailbox as the special instance of folder capable of receiving incoming messages from the mail server; security database for mail user authorization and authentication; and user entry as a location for mail user information.

The BBS provides a forum for sharing ECS related information for users. BBS also permits users to interactively browse and post messages to the bulletin board, download the ECS toolkit, and register with ECS. Through the BBS, users are able to get information from other users, ask questions regarding the information, report concerns or problems, and suggest enhancements. Support of the BBS is provided through the NN package, which allows the application programming interface (API) to post messages to the BB.

The Bulletin Board implementation includes the BB manager to maintain the overall system using interactions such as registering users, toolkit distribution, showing the BB list, on-line help, service statistics, subscriptions, user profiles maintenance, backup to archives, and restore from archives; bulletin board object for message posting, creation, and viewing; message body to provide the capability to read, view, save, and repond to messages; attachment to view files; bulletin board statistics as data statistics for the BB object; and user statistics as data statistics for the users.

The E-Mail Service must be tested to demonstrate the following functions:

- accept an input string for “Subject”
- accept mail addresses
- accept carbon copy (CC) addresses
- accept blind carbon copy (BCC) addresses
- accept addresses for “Reply To”
- accept a message text as a message body
- accept attached files
- deliver message(s) to the described address(s)

The BB management interface must be tested to demonstrate the following:

- accept destination group names (addresses) on which the messages will be posted
- accept subject field
- accept summary for message posted
- accept distribution for message posted
- accept the message body (text)
- accepts attached files which appended on the message body
- post the message on distribution group

Also, tested will be the hypertext links capability and a feature for "WHAT's NEW", which informs the user of new information available on the BBS.

4.11.1.1 Test Case 1: Multi-purpose E-Mail Messaging (TC032.001)

The Multi-purpose E-Mail messaging test demonstrates the capability of the electronic mail messaging function to send and receive the Multi-purpose Internet Mail Extension (MIME) messages based on SMTP/ X.400 protocols existing at GSFC. The MIME extensions help users identify what data type messages they are receiving and produce indications of incompatibilities for readability. Electronic mail written in text can be read by many different machines, but greater data types like graphics could cause readability problems. This test demonstrates the E-Mail capability to automatically respond to messages being received by the DAAC utilizing the "Reply To" object. The response must go to the author and all other incoming destination addresses arriving from the mail server.

Test Configuration:

Hardware: HP and SUN workstation, DCE Cell Configuration, Mail Server

Software: Multi-purpose Internet Mail Extensions (MIME), CsEmMailRelA, testdriver: main.cxx, libCsEmMailRelA

Data: External Generated Data (i.e. user information: addresses, message body, attached files)

Tools: SMTP (RFC 821) /X.400 protocols, system command: 'sendmail'

Test Input:

Test input requires creating two simple mail messages. One will include a message designed to generate an automatic response from the receiving DAAC, which responds only to the author of the message. The second message is designed to distribute an automatic response from the receiving DAAC to a list of destinations addresses listed in the incoming message. These messages must include message body and attached files for complete verification. These

messages will be sent and exchanged with external mail systems based on the internet standard e-mail and the protocols used for the test.

Test Output:

The expected results of this test include successful message transfers across the e-mail gateway. The transport mail program, 'sendmail', is utilized to reply to the author and the destination addresses in the incoming message.

Success Criteria:

This test case is successful if the messages are received and transported successfully back to the sender. Also, the system command, 'sendmail', is capable of interpreting the internet address and relay the electronic message to originator (s).

Test Procedures:

Test Case ID: TC032.001	
Test Name: Multi-purpose E-Mail Messaging	
Test Steps:	Comments:
1. Create an E-Mail message which requires an automatic response to the author only.	Login to the E-mail system; submit a password if required
2. Send message to a recipient at the DAAC.	
3. Verify receipt of proper E-Mail response and destination.	If e-mail address is incorrect, the e-mail will be returned to the sender. Review e-mail addresses, time and the dates Satisfies C-CSS-61010, C-CSS-61030
4. Create an E-Mail message which requires an automatic response from destination addresses	
5. Send message to destination list	
6. Verify receipt of proper E-Mail response and destinations. E-mail messages located: **c d /var/spool/mail/**<username>	
6. Verify that a MIME-type E-Mail message can be sent and received by the tester by creating the message, sending it to an ECS user and requesting return of the original message.	satisfies C-CSS-61020
7. Verify the ability to provide translation between SMTP and X.400 protocols by creating a message in one protocol and sending/receiving it within another.	
8. Go to email directory to run unit test: **cd /ecs/formal/CSS/REL_A/DOF/src/ EMAIL/test/sun5**	

9. Execute the testdriver: **main_sun5**	
10. Verify script ran. Check log.	Satisfies: C-CSS-61290

4.11.1.2 Test Case 2: E-MAIL Mailtool (TC032.002)

The E-Mail Mailtool test demonstrates the capability of the messaging service to manage and interact with user E-mail. Specifically, this test ensures that the E-Mail Mailtool Service is capable of:

- providing a MAILBOX location for incoming messages to be stored and defined folders as archive for users to store data over long periods of time
- providing a MAILBOX area must have a summary of messages containing the author, message title, and date/time of the message origination
- providing the functions to go to next or previous messages, search for keywords within the messages or titles, and search for specific authors in the MAILBOX
- providing a editor, message body, public mailing list, and private mailing list.
- providing the capability to send messages to multiple destinations of type single user, position and site, attaching text or binary files, forwarding messages, discard messages without saving, and search messages for keywords
- providing users of the e-mail system the capability to perform these operations of the editor: copy, delete, cut, paste, undo
- providing an access control feature which requires authentication for access to the Mailtool (BBS) via login and password.

Test Configuration:

Hardware: HP and SUN workstation, DCE Cell Configuration,

Software: Multi-purpose Internet Mail Extensions (MIME), CsEmMailRelA, testdriver: main.cxx, libCsEmMailRelA

Data: External Data (i.e. user information: addresses, message body,attached files)

Tools: SMTP (RFC 821) /X.400 protocols, system command: ‘sendmail’

Test Input:

Test inputs include keyins to create the following:

- MAILBOX which requires authentication for access to the Mailtool via login and password.
- MAILBOX which contain the complete mailtool layout which permits operator interactions with the e-mail system

- MAILBOX which contain radio control buttons to perform mailtool operations
- Messages whose destinations are the bulletin board.

Test Output:

The expected output of this test is to successfully demonstrate the required interactive user functions provided by the MAILBOX and folder. Validation and verification of each test message must be completed. Also, incoming messages are logged to the MAILBOX and folder.

Success Criteria:

This test case is successful when the validation and verification of all Mailtool functions have satisfied the overall test requirements.

Test Procedures:

Test Case ID: TC032.002	
Test Name: E-Mail Mailtool	
Test Steps:	Comments:
1. Verify the existence of e-mail system which has a MAILBOX and folder	satisfies: C-CSS-61310, C-CSS-61330
2. Confirm the mailtool's folder area allows messages to be copied or moved to other operator specified locations	satisfies: C-CSS-61330
3. Validate that folders can store information for long periods of time	satisfies: C-CSS-61320
4. Verify the MAILBOX incorporates: a. subject area b. message body c. destination address area (date/time of message origination) d. radio control buttons e. layout of resources(tool bar, folder status, output area, message summaries, command line, and status bar) f. private mailing lists with one or more destination address	satisfies: C-CSS-61360, C-CSS-61380, C-CSS-61420
5. Perform mail tool functions: a. create/compose a message through the editor. Perform other editor functions: cut, copy, paste,delete, and undo operations. b. attached files: text or binary c. forward a message d. discard a message without saving e. search for keywords in the messages f. go to next or previous message in the MAILBOX g. search MAILBOX or folder for specific names (author) and keywords in title text	satisfies: C-CSS-61370, C-CSS-61430, C-CSS-61440, C-CSS-61450, C-CSS-61460, C-CSS-61470, C-CSS-61490, C-CSS-61500, C-CSS-61510,

4.11.1.3 Test Case 3: E-MAIL API (TC032.003)

The E-Mail API test demonstrates the capability of the E-Mail Service to provide non-interactive messaging services. Specifically, this test ensures that the E-Mail Service is capable of:

- sending an E-Mail message non-interactively from an application
- attaching multiple text or binary files to the mail message.
- accepting a file name as input for the message text.

Test Configuration:

Hardware: HP and SUN workstation, DCE Cell Configuration

Software: Multi-purpose Internet Mail Extensions (MIME), Usernet BBS viaXRN interface, CsEmMailRelA, testdriver: main.cxx, libCsEmMailRelA

Data: External Data (i.e. user information: addresses, message body,attached files)

Tools: SMTP (RFC 821) /X.400 protocols, system command: 'sendmail'

Test Input:

Test input requires an API to be written to demonstrate the capability to send an E-Mail message programmatically. The message sent must have multiple files attached to it and have the file name as input for the message text. This message, along with the attachment, is then sent to multiple destinations, including a bulletin board, and the E-Mail Service must accept the mailing list as valid destinations.

Test Output:

The expected outputs of this test include a successful demonstration of all the required non-interactive user functions provided by E-Mail API. Validation and verification of each test message must be complete and satisfactory.

Success Criteria:

This test is successful when validation and verification of all API functions have been determined to satisfy the overall test requirements.

Test Procedures:

Test Case ID: TC032.003	
Test Name: E-Mail API	
Test Steps	Comments:
1. Create a message with an attached binary file to go to multiple destinations	

2. Initiate the the sytem command sendmail. **cd /usr/lib** **sendmail** <destination addresses> Enter text(mesage) attach binary **send mail**	satsfies: C-CSS-61390, C-CSS-61800, C-CSS-61810, C-CSS-61840,
3. Verify message sent and time	satisfies: C-CSS-61800. C-CSS-61810, C-CSS-61840
4. Create a message with an attached text files to go to multiple destinations	
5. Repeat step 2	satisfies: C-CSS-61810
6. Verify the mailtool API accepts mailing list identifications (private and public) as valid destinations, accept file name as input for message text, and accept attached files.	satisfies: C-CSS-61850. C-CSS-61820, C-CSS-61520
7. Validate that the mailtool API permits destinations of single, position of one or more operators, and site of one or more operators	satisfies: C-CSS-61400

4.11.1.4 Test Case 4: General Bulletin Board Service (TC032.004)

The General BBS test demonstrates the capability of the BBS Service to provide a forum for sharing ECS related information on the platforms the service supports (TCP/IP, NNTP, SMTP, and Usernet message standards), to host the user registration service, and to download ECS toolkits.

The General BB provides users with the xrn public browser for interactive functionalities. For news data, the Usernet via xrn interface, uses the NNTP remote news server for accessing news groups and articles on the server, thus, allowing users to read from the central news repository.

Specifically, this test ensures that the BBS Service is capable of:

- supporting multiple bulletin board and multiple messages for each bulletin board.
- providing concurrent access to multiple users.
- allowing authorized users to create new bulletin board, delete existing bulletin board, delete
- message(s) from a bulletin board, force users off bulletin boards for backup, collect access
- history and/or statistical information.
- providing users with new information available on the bulletin boards.

Test Configuration:

Hardware: HP and SUN workstation, DCE Cell Configuration, NNTP server

Software: Multi-purpose Internet Mail Extensions (MIME), Usernet BBS viaxrn interface (public browser), CsBBMailRelA, testdriver: bbmain.cxx,

Data: External Data, user information (i.e. addresses, message body,attached files)

Tools: SMTP (RFC 821) /X.400, TCP/IP and NNTP protocols, NN package

Test Input:

The test input requires destination addresses, message body, and attached files.

Test Output:

The expected output includes user information and the capability of the Bulletin Board Service to provide a proper forum for sharing ECS related information on the required platforms and interactive functions. The BBS must also demonstrate the capabilities to download ECS toolkits as specified.

Success Criteria:

This test is successful if the BB service allows user registration and the support to download ECS toolkits. Also, this test is successful when validation and verification of all API functions have been determined to satisfy the overall test requirements.

Test Procedures:

Test Case ID: TC032.004	
Test Name: General Bulletin Board Service	
Test Steps:	Comments:
1. Bring up the USERNET news system via the XRN interface.	
2. Login users.	
3. Verify multiple messages can be posted for various bulletin boards.	
4. Verify multiple bulletin boards accessed via XRN interface.	
5. Create a new bulletin board message and annotate an audio file and a video file within the message.	
6. Access a secured bulletin board from the BBS Service and verify that user registration is required.	Satisfies: C-CSS-62050
7. Login to the secured bulletin board.	
8. Create a file with specified permissions, which grant only authorized users access to the file.	
9. Attempt to access the transferred file, with and without the appropriate access controls.	
10. Verify the access controls transferred with the file.	

11. Create a statistical report of the access history which includes the following information: a. each user b. system login count c. total time on-line d. total time for downloads e. messages read status for each BBS f. each BBS g. message count h. last cleanup i. last backup j. access count	satisfies: C-CSS-62080
12. Create an ECS toolkit distribution list.	
13. Distribute the toolkit.	
14. Verify that the toolkit distributed properly.	Satisfies: C-CSS-62070
15. Using the BBS management interface execute the following functions:	
a. creating new BB b. deleting existing BB c. deleting messages(s) from BB d. backup BB(s) e. force users off a single BB or entire system for backup f. collecting access history and/or statistical information	
16. Verify all options listed function properly.	
17. Access the online help facility.	
18. Verify help function is accurate.	
19. Create a message reply response to the author of a bulletin board message.	
20. Post a message response to multiple bulletin boards.	
21. Post a message response to a single bulletin board.	
22. Access the BBS news update feature.	
23. Explore the hyperlink sublevels and check for consistency within the operations and the menus..	
24. Verify the existence of the “WHAT’s NEW” option	satisfies: C-CSS-62130
25. Access the “WHAT’s NEW” option and gather information. Save data and place in archive	

4.11.1.5 Test Case 5: Mailtool Bulletin Board Service (TC032.005)

The BBS Mailtool test demonstrates the capability of interactive functionality for any bulletin board user. The bulletin board is tested for the capability of users to either subscribe or unsubscribe to any bulletin boards and to select a subscribed bulletin board for viewing a summary of all messages in that bulletin board. The capability of the bulletin board to respond to a posted message by sending the response to the bulletin board and/or to the author and/or any named destinations and/or any other operator specified destination is also tested.

The following search capabilities are verified for bulletin board users:

- search for a string in message headers and in message text
- search by author
- search by subject

A catch-up feature which excludes user specified messages from appearing in the bulletin board is also tested. The bulletin board services must demonstrate that users and applications can post messages to bulletin board(s) and collect, maintain, and access history and statistical information for the service. Copying/saving messages to the local bulletin board system is tested along with attaching ASCII or binary files to a message being sent.

The BBS must demonstrate the following bulletin board configuration options:

- screen size
- number of messages displayed on a screen
- screen colors (background/foreground)
- read message indicator

Test Configuration:

Hardware:	HP and SUN workstation, DCE Cell Configuration, NNTP server
Software:	Multi-purpose Internet Mail Extensions (MIME), Usernet BBS viaXRN interface, CsBBMailRelA, testdriver: bbmain.cxx,
Data:	External Data, user information(i.e.: addresses, message body,attached files)
Tools:	SMTP (RFC 821) /X.400, TCP/IP, and NNTP protocols, NN package

Test Input:

Sequence of interactive commands exercising BBS usage for search, configuration, message retrieval, copying files, and attaching files to a message. Also, input includes the following for postage: destination group names, subject, keyword, message summaries, and distribution for message. The message body and attached files are also BB targets of input for test.

Test Output:

The expected results of these tests are a demonstration of the full capabilities of the BBS Service. Specifically, the capabilities to maintain access history and statistical information, copying files, and attaching ASCII or Binary files.

Success Criteria:

This test case will be successful if the messages and files on destination groups are posted. Also, success criteria for BBS functions have been determined to satisfy the overall test requirements.

Test Procedures:

Test Case ID: TC032.005	
Test Name: MailTool Bulletin Board Service	
Test Steps:	Comments:
1. Copy files from the BB.	
2. Demonstrate the functionality of the Bulletin Board Service to collect and maintain access history.	
3. Examine the statistical information service. Validate the BBS capability to maintain the history and statistical information by adding new data and retrieving old data	satisfies: C-CSS-62080
4. Verify that you can access the Bulletin Board Service in interactive mode from the command line.	
5. Subscribe and unsubscribe to the BB through the Bulletin Board Service.	
6. Select a subscribed BB to view summary information of all the messages within that BB.	
7. Respond to a message specifying the destination.	
8. Demonstrate the capability to perform a search by various topics (i.e., author, subject, and keyword).	
9. Demonstrate the functionality of the catch-up feature.	Satisfies: C-CSS-62060
10. Verify that the BB is capable of copying files.	Satisfies: C-CSS-62060
11. Verify the interface that exists to allow applications to post a message to the bulletin board. Demonstration BB initialization through NN package.	Satisfies: C-CSS-62800
12. Create a message with an attachment.	
13. Verify users can attach ASCII and Binary files.	C-CSS-62390

4.11.1.6 Test Case 6: API Bulletin Board Service (TC032.006)

The BBS API test will demonstrate the capability of the BBS Service to provide non-interactive BBS Services for programmatic interface functions along with attaching ASCII and/or binary files to a message being sent. The capability to post a message to multiple bulletin boards will also be verified.

Test Configuration:

Hardware: HP and SUN workstation, DCE Cell Configuration, NNTP server

Software: Multi-purpose Internet Mail Extensions (MIME), Usernet BBS viaXRN interface, CsBBMailRelA, testdriver: bbmain.cxx,

Data: External generated Data, user information (i.e. addresses, message body,attached files)

Tools: SMTP (RFC 821) /X.400, TCP/IP, NNTP protocols, NN package

Test Input:

Test inputs include demonstrating the capability of the BBS Service to provide a programmatic interface to post a message to a bulletin board and to have a message posted to multiple bulletin boards. Also, the BBS will be tested for attaching ASCII and binary files to a message.

Test Output:

The expected results of these tests are a successful demonstration of the capability of the BBS Service to provide non-interactive functions to BBS users. Also, the verification of the interface for the applications to post a message to a bulletin board and multiple bulletin boards.

Success Criteria:

This test case is successful when validation and verification of all BBS functions have been determined to satisfy the overall test requirements.

Test Procedures:

Test Case ID: TC032.006	
Test Name: API Bulletin Board Service	
Test Steps:	Comments:
1. Go to the test directory and execute the bbmain_sun test driver	
2. Verify the bulletin board accepts: a). destination group names b). subject field c). key word for message posted d). message summary e). distribution for message posted f). message text g). attached files (ASCII and Binary)	satisfies: C-CSS-62810
3. Demonstrate the capability to post a message to multiple BB.	satisfies: C-CSS-62820
4. Create a message to be posted	
5. Post message to BB	
6. Create a message with ASCII attachment	
7. Post message with attachment	
8. Create message with binary attachment	
9. Post message with attachment	
10. Verify messages were complete as written with attachment(s).	
Create a message to be posted	

4.11.2 Virtual and X-Terminal Support Thread (TC033)

The purpose of this thread is to verify that users and the server host can access ECS host using a virtual device. Terminal characteristics and handling conventions are hidden in use. The Virtual Terminal provide users with remote logins. The other objective of this thread is to verify that the Communication Subsystem (CSS) Common Facilities provide a standard graphical user interface (GUI) that can be used for remote GUI based sessions from various platforms as well as users

within and outside of the ECS domain. This thread verifies the graphical user interface for the following applications:

- Tools standards: telnet(remotely), kerberized version of telnet protocol
- CSS Common Facilities: File Access and Transfer, Bulletin Board, Electronic
- Mail, and remote terminal support

Special resources required for this thread test include:

- Operational DCE cell
- Host workstations with X-window capability
- Simulated access to workstations at DAACs, SCFs, and External ECS Users

Additional resource requirements are specified as required for each test case.

4.11.2.1 Test Case 1: Secure/Non-Secure Virtual Terminal (TC033.001)

The purpose of the Secure Virtual Terminal test is to verify that client server terminal interaction can be handled through a virtual device with common capabilities, thus hiding the underlying terminal characteristics from the user. This capability is provided to DAAC and SCF users within/outside the ECS domain through a secured and non-secured virtual terminal service (telnet).

Test Configuration:

Hardware: Operational DCE Cell, Host Workstations with X-window capability,Virtual Teminal Server

Software: Operating system software

Data: Extracted data

Tools: Simulated access to workstation at DAACs, SCFs, and ECS users

Test Input:

Inputs include internal and remote host connectivity to establish virtual terminal sessions within/outside the DAAC and messages sent via command line.

Test Output:

The expected results of this test include terminal sessions to remote DAAC/SCF hosts and the execution of commands received from the remote service.

Success Criteria:

This test is successful when users are able to connect through remote virtual terminal sessions to DAAC and SCF hosts.

Test Procedures:

Test Case ID: TC033.001	
Test Name: Secure Virtual Terminal	
Test Steps:	Comments:
1. Login to local DAAC host.	
2. Telnet over to a remote DAAC host.	
3. Login to the ECS guest server, as a non-registered user with guest privileges	
4. Verify that the current terminal allows access to a remote DAAC host (directories, files, etc.).	satisfies: C-CSS-63050
5. Verify that the means to enhance characteristics of the basic virtual device is by mutual agreement between the two communicating parties.	
6. Verify that the virtual terminal is capable of supporting octet.	
7. Logout of remote DAAC host.	
8. Repeat steps 1 through 4 for the following combinations:	
a. SCF to DAAC	
b. DAAC to SCF	
c. SCF to SCF	
9. Verify VT service can connect, login, send command, receive command, and update display	satisfies: C-CSS-63050
10. Verify common facilities: e-mail, bullentin board, remote access, and file transfer applications	
11. Demonstrate operating system utilities and various x-terminal support.	satisfies: C-CSS-63060

4.11.2.3 Test Case 2: Common Facility GUI Based Sessions (TC033.002)

The purpose of the Common Facility GUI Based Sessions test is to demonstrate the graphical user interface for the various tools provided with the CSS Common Facilities and Services.

Test Configuration:

Hardware: Operational DCE Cell, Host Workstations with X-window capability

Software: Operating system software

Data: External generated data

Tools: Simulated access to workstation at DAACs, ECS users,

Test Input:

Inputs include actions to exercise the CSS common facilities.

Test Output:

The expected results of this test include successful demonstration of CSS Common Facilities and Services.

Success Criteria:

This test case will be successful if all CSS Common Facilities and Services operate through GUIs.

Test Procedures:

Test Case ID: TC033.002	
Test Name: Common Facility GUI Based Sessions	
Test Steps:	Comments:
1. Verify X-applications through the operating system software	satisfies: C-CSS-63060
2. Verify file access and transfer service, bulletin board service, electronic mail service, and remote virtual service threads	Entire communications infrastructure thread verified

4.11.3 Managed ftp Thread (TC034)

The purpose of this thread is to verify that the CSS File Access service supports the file transfer protocol (FTP) functionality and the kerberized file transfer protocol (KFTP). FTP is used to send and receive files on a network. A program is run on the client machine which connects to a process on a server machine. The client may then send and receive files. KFTP is used in the same context as FTP but it includes higher security measures.

This thread must also assist the security interests associated with kerberized ftp. Kerberized ftp provides a layer of authentication to ftp, whereas, entities communicating over networks have to prove their identities. In broader terms, specific credentials are required, and those who do not have the correct credentials will not be permitted to use the 'kftp' application. There is a kerberized server set to maintain a database of user, server, and password information.

Kerberos is a network authentication system for use on physically insecure networks. Kerberos works by providing principals (users or services) with tickets that they can use to identify themselves to other principals and secret cryptographic keys for secure communication with other principals. A ticket is a sequence of a few hundred bytes that can be embedded in a network protocol. thereby allowing the processes implementing that protocol to validate the identity of the principals involved. Kerberos also provides data stream integrity (detection of modification) and secrecy (preventing unauthorized reading) using cryptography systems such as DES (data encryption standard).

Testing will consist of:

- sending files
- receiving files
- transferring files via a proxy account
- event logging
- batch file transfer

4.11.3.1 Test Case 1: Sending Files via FTP (TC034.001)

This test verifies the ability to send files to local and remote machines via FTP.

Test Configuration:

Hardware: SunOS 5.4, HPUX9, SGI IRIX5

Software: CsFtFTPRelA, test_PH3

Data: Internal/External data files

Tools: ftp API, Rogue Wave 6.1, testdriver: ftptest

Test Input:

A binary or text file retrieved from a local DAAC host via ftp.

Test Output:

FTP transfer status logs. Also, the created file: libCsFtFTPRelA.a.

Success Criteria:

The files to be transferred are moved to the local and remote machines.

Test Procedures:

Test Case ID: TC034.001	
Test Name: Sending Files via FTP	
Test Steps:	Comments
1. Login to local host. Remotely login to another machine: **rlogin** <host>; password Spawn an independent FTP client process. **ftp** <host> username and password	
2. At the FTP prompt, attempt to send a file to another machine. **ls -al** **put** <filename>	places file at specifed location if permitted
3. In ftp mode: **!ls -al**	verify file sent of same size
4. **bye**	exits ftp session.

5. Verify that the other machine received the file by looking into its directory structure. **ls -al**	
6. Verify that the file received is the same as the file sent by invoking the diff command. **diff**<filename> <filename>	Two identical files names are compared in file size
7. Repeat this scenario with multiple files. Remote login. **rlogin** <host> Open another ftp session: **ftp** <host> ;username and password	satisifies: C-CSS-60900
8. **status**	provides information on proxy connection, mode, file type,form structure, verbose, sound, prompt and glob mode, overwrite monitors (store and receive), transmission boolean, n-ampping boolean, hash marking, and use of port commands . file type default : ASCII; satisfies: C-CSS-60920
9. **type**	signifies current type
10. User selects type binary or ASCII: type **binary**; type set to binary (I) type **ascii**; type set to ASCII	Type is either ASCII or Binary; satisfies: C-CSS-60910
11. **mput** <file1> <file2> <file3> <file4> <file5>	put multiple files on remote machine
12. **!ls -al**	verify file sizes of the five files
13. **bye**	exits ftp session
14. Verify that the events were logged correctly in the appropriate log files. This should be checked for single and multiple file transfers.	

4.11.3.2 Test Case 2: Transferring Files via a Proxy Account (TC034.002)

A proxy machine is one that answers requests for another machine. By simulating its identity, the machine accepts responsibility for transferring packets intended for the real destination. This test verifies the ability to send files to local and remote machines via FTP using a proxy account.

Test Configuration:

Hardware: SunOS 5.4, HPUX9, SGI IRIX5

Software: CsFtFTPRelA, test_PH3

Data: Internal/External data files

Tools: ftp API, Rogue Wave 6.1, testdriver: ftptest

Test Input:

A binary or text file is extracted from the local and remote terminals.

Test Output:

FTP transfer status logs. A file is transferred to the specified location.

Success Criteria:

The files are transferred from one remote host to another by the proxy account.

Test Procedures:

Test Case ID: TC034.002	
Test Name: Transferring Files via a Proxy Account	
Test Steps:	Comments:
1. Log onto a remote host via telnet. **telnet**<host>; login and password	
2. Spawn an independent FTP client process. **ftp** <host1>; user name and password	
3. Have the two remote hosts transfer a file between each other.	
4. In ftp mode: **status**	verifies no proxy connection
5. Perform: **proxy open** <host2> status	verifies proxy connection to host2
6. Do: **get** <filename> (from host2)	
7. **!ls -al**	verify transfer on host 1
8. **close**	
9. Do: **get** <filename> (from host 1)	
10.**bye**	
11.**ls -al**	verifies transfer on telnet connection
12.**exit**	exits telnet connection
13. Verify that the receiving machine does not have any files transferred to its directory structure. **ls -al**	Files were not transferred. Verify last update
14. Telnet to host machine **telnet**<host> Verify that the file received is the same as the file sent by invoking the diff command. diff <file1> <file2>	
15. Repeat this scenario with multiple files. **telnet** <host>; provide login and password	Telnet session starts
16. **ftp** <username>; password	
17. **ls -al**	current listings
18 **mget**<file1><file2><file3><file4>	transfer multiple files
19 **!ls -al**	verify files of same sizes are transferred
20 **close**< remote host name>	
21 **mget** <file1><file2><file3><file4>	
22. **!ls -al**	
23. **bye**	exits ftp session
24. **exit**	exits ftp session
25. Verify that the events were logged correctly in the appropriate log files. This should be checked for single and multiple file transfers. **cd /ecs/formal/CSS/REL_A/DOF/src/FTPSCHED/test/phase3**	
26. Execute: **ftpschedTest_sun5**	partially satisfies: C-CSS-60800

27. Go to designated file transfer log areas and verify have been transferrred to correct data locations **cd** <data> **ll** <filename>	The actual move is denoted. Contents of the files are verified. satisfies: C-CSS-60800, C-CSS-60810, C-CSS-60820

4.11.3.3 Test Case 3: Receiving Files via FTP (TC034.003)

This test verifies the ability to receive files from local and remote machines via FTP.

Test Configuration:

Hardware: SunOS 5.4, HPUX9, SGI IRIX5

Software: CsFtFTPRelA, test_PH3

Data: Internal/External data files

Tools: ftp API, Rogue Wave 6.1, testdriver: ftptest

Test Input:

A binary or text file is transferred from one host machine to another host machine

Test Output:

Files are sent and received between hosts. FTP transfer status logs.

Success Criteria:

The files are received successfully on the local machine.

Test Procedures:

Test Case ID: TC034.003	
Test Name: Receiving Files via FTP	
Test Steps:	Comments:
1. Spawn an independent FTP client process. **ftp** <host> username and password	
2. Verify capabilty to utilize the ftp API.	satisfies: C-CSS-60900
3. In ftp mode: **status**	satisifes: C-CSS-60920
2. At the FTP prompt, attempt to receive a file to another machine. **get**<filename>	
3. Verify that the local machine received the file by looking into its directory structure. **!ls -al**	
4. Repeat this scenario with multiple files. **mget**<file1><file2> <file3> <file4> **!ls -al**	Verifies files transferred to original host
6. Type: **bye**	exits ftp mode
5. Look in current directory for newly extracted files from alternate local host. **ls -al**	
6. Repeat steps 1-5 for remote machines.	

4.11.3.4 Test Case 4: Batch File Transfers (TC034.004)

This test verifies the ability to receive and send files from local and remote machines via FTP in batch mode.

Test Configuration:

Hardware: SunOS 5.4, HPUX9, SGI IRIX5

Software: CsFtFTPRelA, ftpSchedTest.C, kftpSchedTest.C

Data: Internal/External data files

Tools: ftp API, kftp API, Rogue Wave 6.1, testdrivers: ftpSchedTest andkftpSchedTest

Test Input:

ASCII or binary files are sent and received successfully via ftp and kftp in batch mode.

Test Output:

Files are transferred via ftp and kftp. FTP/KFTP transfer status logs show files were transferred to their designated locations.

Success Criteria:

The batch files are received successfully on the local machine. Events were logged to the appropriate designated locations.

Test Procedures:

Test Case ID: TC034.004	
Test Name: Batch File Transfers	
Test Steps:	Comments:
1. Login to local DAAC host	
2. Attempt to send a file to another machine by submitting a ftp batch process. **cd /ecs/formal/CSS/REL_A/DOF/src/FTPSCHED/test/phase3**	
3. Execute unit test for batch ftp: **ftpSchedTest_sun5**	For the SUN
4. Verify that the files were sent and received by the host machine at the designated time by checking the contents of the directory structures. **ls -al here***	Data is in 'here directories: here, here1, here2, here3, here5 Notice: 'here4' is an intentional missing directory for error checking satisfies: C-CSS-60810
5. Type: **pwd**	verify current directory

6. Attempt to send a file to another machine by submitting a kftp batch process. **cd /ecs/formal/CSS/REL_A/DOF/src/FTPSCHED/test/phase3** Execute unit test for batch kftp: **kftpSchedTest_sun5**	
7. Verify that the files were sent and received by the host machine at the designated time by checking the contents of the directory structures. **ls -al here***	Data is in 'here directories: here, here1, here2, here3, here5 Notice: 'here4' is an intentional missing directory for error checking; satisfies: C-CSS-60530
8. Go into DCE: **dce_login**; provide principal name and password	
9. Type: **klist**	list kerberized credentials for verification
10. Do: **knit** <username>; password	get kerberos ticket(s)
11. Enter: **cd /krb5/kftpdir**	
12. Perform: **kftp** <host>; username and password	
13. Perform a simple file transfer using kerberos: **get** <filename>	
14. **!ls -al**	verify the transfer in kftp mode satisfies: C-CSS-60530
15. **bye**	exit kftp
16. **kdestroy**	kill kerberos ticket information
17. Verify Steps 1-5 on HP and SGI platforms	For HP and SGI

4.11.4 Communications Infrastructure 3 Build (BC010)

The Communications Infrastructure 3 Build consists of integrating and testing the threads and build testcases in order to demonstrate their functionality, first as an integrated sub-component, with all the functionality built together and then to demonstrate the functionality implemented and operating at the system level from end-to-end. The following threads and build will be demonstrated:

- E-Mail/BBS Thread
- Virtual and X-terminal Support Thread
- Managed ftp Thread
- Gateway Client/Server Thread
- Communications Infrastructure 2 Build

4.11.4.1 Test Case 1: Communication Infrastructure 2 Regression Test (BC010.001)

This test is a regression test of the overall functionality of the Communication Infrastructure 2 build while integrated as part of the Communications Infrastructure 3 build. The objective of this

test is to verify that all of the Communication Infrastructure 2 functionality remains available and unchanged upon integration of the Communication Infrastructure 3 Build as when demonstrated as a single subsystem component which included the Communication Infrastructure 1 build integrated with the Message Passing 2 extended functionality. This regression includes the Communications Infrastructure 2 level 3's.

4.11.4.2 Test Case 2: Multi-purpose E-Mail Messaging (BC010.002)

The Multi-purpose E-Mail messaging test demonstrates the capability of the electronic mail messaging function to send and receive the Multi-purpose Internet Mail Extension (MIME) messages based on SMTP/ X.400 protocols existing at GSFC. The MIME extensions help users identify what data type messages they are receiving and produce indications of incompatibilities for readability. Electronic mail written in text can be read by many different machines, but greater data types like graphics could cause readability problems. This test demonstrates the E-Mail capability to automatically respond to messages being received by the DAAC utilizing the "Reply To" object. The response must go to the author and all other incoming destination addresses arriving from the mail server.

Test Configuration:

Hardware: HP and SUN workstation, DCE Cell Configuration, Mail Server

Software: Multi-purpose Internet Mail Extensions (MIME), CsEmMailRelA, testdriver: main.cxx, libCsEmMailRelA

Data: External Generated Data (i.e. user information: addresses, message body,attached files)

Tools: SMTP (RFC 821) /X.400 protocols, system command: 'sendmail'

Test Input:

Test input requires creating two simple mail messages. One will include a message designed to generate an automatic response from the receiving DAAC, which responds only to the author of the message. The second message is designed to distribute an automatic response from the receiving DAAC to a list of destinations addresses listed in the incoming message. These messages must include message body and attached files for complete verification. These messages will be sent and exchanged with external mail systems based on the internet standard e-mail and the protocols used for the test.

Test Output:

The expected results of this test include successful message transfers across the e-mail gateway. The transport mail program, 'sendmail', is utilized to reply to the author and the destination addresses in the incoming message.

Success Criteria:

This test case is successful if the messages are received and transported successfully back to the sender. Also, the system command, 'sendmail', is capable of interpreting the internet address and relay the electronic message to originator (s).

Test Procedures:

Test Case ID: BC010.002	
Test Name: Multi-purpose E-Mail Messaging	
Test Steps:	Comments:
1. Create an E-Mail message which requires an automatic response to the author only.	Login to the E-mail system; submit a password if required
2. Send message to a recipient at the DAAC.	
3. Verify receipt of proper E-Mail response and destination.	If e-mail address is incorrect, the e-mail will be returned to the sender. Review e-mail addresses, time and the dates C-CSS-61010, C-CSS-61030
4. Create an E-Mail message which requires an automatic response from destination addresses	
5. Send message to destination list	
6. Verify receipt of proper E-Mail response and destinations. E-mail messages located: **c d /var/spool/mail/<**username>	
7. Verify that a MIME-type E-Mail message can be sent and received by the tester by creating the message, sending it to an ECS user and requesting return of the original message.	C-CSS-61020
8. Verify the ability to provide translation between SMTP and X.400 protocols by creating a message in one protocol and sending/receiving it within another.	ESN-0010#A(a, b, c, d, e, f, g, h) EOSD0500#A(d, e, f, g, h, I) ESN-0340#A, ESN0345#A, ESN-0350#A
9. Go to email directory to run unit test: **cd /ecs/formal/CSS/REL_A/DOF/src/ EMAIL/test/sun5**	
10. Execute the testdriver: **main_sun5**	
11. Verify script ran. Check log.	C-CSS-61290

4.11.4.3 Test Case 3: E-MAIL Mailtool (BC010.003)

The E-Mail Mailtool test demonstrates the capability of the messaging service to manage and interact with user E-mail. Specifically, this test ensures that the E-Mail Mailtool Service is capable of:

- providing a MAILBOX location for incoming messages to be stored and defined folders as archive for users to store data over long periods of time

- providing a MAILBOX area must have a summary of messages containing the author, message title, and date/time of the message origination
- providing the functions to go to next or previous messages, search for keywords within the messages or titles, and search for specific authors in the MAILBOX
- providing a editor, message body, public mailing list, and private mailing list.
- providing the capability to send messages to multiple destinations of type single user, position and site, attaching text or binary files, forwarding messages, discard messages without saving, and search messages for keywords
- providing users of the e-mail system the capability to perform these operations of the editor: copy, delete, cut, paste, undo
- providing an access control feature which requires authentication for access to the Mailtool (BBS) via login and password.

Test Configuration:

Hardware:	HP and SUN workstation, DCE Cell Configuration,
Software:	Multi-purpose Internet Mail Extensions (MIME CsEmMailRelA, testdriver: main.cxx, libCsEmMailRelA
Data:	External Data (i.e. user information: addresses, message body, attached files)
Tools:	SMTP (RFC 821) /X.400 protocols, system command: 'sendmail'

Test Input:

Test inputs include keyins to create the following:

- MAILBOX which requires authentication for access to the Mailtool via login and password.
- MAILBOX which contain the complete mailtool layout which permits operator interactions with the e-mail system
- MAILBOX which contain radio control buttons to perform mailtool operations
- Messages whose destinations are the bulletin board.

Test Output:

The expected output of this test is to successfully demonstrate the required interactive user functions provided by the MAILBOX and folder. Validation and verification of each test message must be completed. Also, incoming messages are logged to the MAILBOX and folder.

Success Criteria:

This test case is successful when the validation and verification of all Mailtool functions have satisfied the overall test requirements.

Test Procedures:

Test Case ID: BC010.003	
Test Name: E-Mail Mailtool	
Test Steps:	Comments:
1. Verify the existence of e-mail system which has a MAILBOX and folder	C-CSS-61310, C-CSS-61330
2. Verify the MAILBOX incorporates: a. subject area b. message body c. destination address area (date/time of message origination) d. radio control buttons e. layout of resources(tool bar, folder status, output area, message summaries, command line, and status bar)	C-CSS-61360, ESN-0010#A (b) EOSD0500#A(d, e, f, g, h, I)

4.11.4.4 Test Case 4: General Bulletin Board Service (BC010.004)

The General BBS test demonstrates the capability of the BBS Service to provide a forum for sharing ECS related information on the platforms the service supports (TCP/IP, NNTP, SMTP, and Usernet message standards), to host the user registration service, and to download ECS toolkits.

The General BBS provides users with the xrn public browser for interactive functionalities. For news data, the Usernet via xrn interface, uses the NNTP remote news server for accessing news groups and articles on the server, thus, allowing users to read from the central news repository.

Specifically, this test ensures that the BBS Service is capable of:

- supporting multiple bulletin board and multiple messages for each bulletin board.
- providing concurrent access to multiple users.
- allowing authorized users to create new bulletin board, delete existing bulletin board, delete
- message(s) from a bulletin board, force users off bulletin boards for backup, collect access

- history and/or statistical information.
- providing users with new information available on the bulletin boards.

Test Configuration:

Hardware: HP and SUN workstation, DCE Cell Configuration, NNTP server

Software: Multi-purpose Internet Mail Extensions (MIME), Usernet BBS viaxrn interface (public browser), CsBBMailRelA, testdriver: bbmain.cxx,

Data: External Data, user information (i.e. addresses, message body,attached files)

Tools: SMTP (RFC 821) /X.400, TCP/IP and NNTP protocols, NN package

Test Input:

Test inputs require the following protocol standards to be configured: SMTP, NNTP and Usernet message standard (RFC 850). The TCP/IP standard usually comes configured within the operating system. Also, the test input requires destination addresses, message body, and attached files.

Test Output:

The expected output includes user information and the capability of the Bulletin Board Service to provide a proper forum for sharing ECS related information on the required platforms and interactive functions. The BBS must also demonstrate the capabilities to download ECS toolkits as specified.

Success Criteria:

This test is successful if the BB service allows user registration and the support to download ECS toolkits. Also, this test is successful when validation and verification of all API functions have been determined to satisfy the overall test requirements.

Test Procedures:

Test Case ID: BC010.004	
Test Name: General Bulletin Board Service	
Test Steps:	Comments:
1. Bring up the USERNET news system via the XRN interface.	
2. Login users.	
3. Verify multiple messages posted for various bulletin boards.	
4. Verify multiple bulletin boards accessed via XRN interface.	
5. Create a new bulletin board message and annotate an audio file and a video file within the message.	

6. Access a secured bulletin board from the BBS Service and verify that user registration is required.	
7. Login to the secured bulletin board.	
8. Create a file with specified permissions, which grant only authorized users access to the file.	
9. Attempt to access the transferred file, with and without the appropriate access controls.	
Verify the access controls transferred with the file.	
Create a statistical report of the access history which includes the following information:	
a. each user b. system login count c. total time on-line d. total time for downloads e. messages read status for each BBS f. each BBS g. message count h. last cleanup i. last backup j. access count	
12. Create an ECS toolkit distribution list.	
13. Distribute the toolkit.	
14. Verify that the toolkit distributed properly.	
Using the BBS management interface execute the following functions:	
creating new BB deleting existing BB deleting messages(s) from BB backup BB(s) force users off a single BB or entire system for backup collecting access history and/or statistical information	
16. Verify all options listed function properly.	
17. Access the online help facility.	
18. Verify help function is accurate.	
Create a message reply response to the author of a bulletin board message.	
Post a message response to multiple bulletin boards.	
Post a message response to a single bulletin board.	
Access the BBS news update feature.	
Explore the hyperlink sublevels and check for consistency within the operations and the menus..	ESN-0010#A (a, c, d, e, f, g, h, i) EOSD0500#A(d, e, f, g, h, I) ESN-1181#A, AM1-0220#A

4.11.4.5 Test Case 5: Mailtool Bulletin Board Service (BC010.005)

The BBS Mailtool test demonstrates the capability of interactive functionality for any bulletin board user. The bulletin board is tested for the capability of users to either subscribe or

unsubscribe to any bulletin boards and to select a subscribed bulletin board for viewing a summary of all messages in that bulletin board. The capability of the bulletin board to respond to a posted message by sending the response to the bulletin board and/or to the author and/or any named destinations and/or any other operator specified destination is also tested.

The following search capabilities are verified for bulletin board users:

- search for a string in message headers and in message text
- search by author
- search by subject

A catch-up feature which excludes user specified messages from appearing in the bulletin board is also tested. The bulletin board services must demonstrate that users and applications can post messages to bulletin board(s) and collect, maintain, and access history and statistical information for the service. Copying/saving messages to the local bulletin board system is tested along with attaching ASCII or binary files to a message being sent.

The BBS must demonstrate the following bulletin board configuration options:

- screen size
- number of messages displayed on a screen
- screen colors (background/foreground)
- read message indicator

Test Configuration:

Hardware: HP and SUN workstation, DCE Cell Configuration, NNTP server

Software: Multi-purpose Internet Mail Extensions (MIME), Usernet BBS viaXRN interface, CsBBMailRelA, testdriver: bbmain.cxx,

Data: External Data, user information(i.e.: addresses, message body,attached files)

Tools: SMTP (RFC 821) /X.400, TCP/IP, and NNTP protocols, NN package

Test Input:

Sequence of interactive commands exercising BBS usage for search, configuration, message retrieval, copying files, and attaching files to a message. Also, input includes the following for postage: destination group names, subject, keyword, message summaries, and distribution for message. The message body and attached files are also BB targets of input for test.

Test Output:

The expected results of these tests are a demonstration of the full capabilities of the BBS Service. Specifically, the capabilities to maintain access history and statistical information, copying files, and attaching ASCII or Binary files.

Success Criteria:

This test case will be successful if the messages and files on destination groups are posted. Also, success criteria for BBS functions have been determined to satisfy the overall test requirements.

Test Procedures:

Test Case ID: BC010.005	
Test Name: MailTool Bulletin Board Service	
Test Steps:	Comments:
1. Copy files from the BB.	
2. Demonstrate the functionality of the Bulletin Board Service to collect and maintain access history.	
3. Inspect the statistical information for the service.	
4. Verify that you can access the Bulletin Board Service in interactive mode from the command line.	
5. Subscribe and unsubscribe to the BB through the Bulletin Board Service.	
6. Select a subscribed BB to view summary information of all the messages within that BB.	
7. Respond to a message specifying the destination.	
8. Demonstrate the capability to perform a search by various topics (i.e., author, subject, and keyword).	
9. Demonstrate the functionality of the catch-up feature.	
10. Verify that you can save/copy a message to the local system.	
11. Test the interface that exists to allow applications to post a message to the bulletin board.	
12. Verify users can attach ASCII and Binary files.	ESN-0010#A (h) EOSD0500#A(d, e, f, g, h, I)

4.11.4.6 Test Case 6: Secure/Non-Secure Virtual Terminal (BC010.006)

The purpose of the Secure Virtual Terminal test is to verify that client server terminal interaction can be handled through a virtual device with common capabilities, thus hiding the underlying terminal characteristics from the user. This capability is provided to DAAC and SCF users within/outside the ECS domain through a secured and non-secured virtual terminal service (telnet).

Test Configuration:

Hardware: Operational DCE Cell, Host Workstations with X-window capability,Virtual Terminal Server

Software: Operating system software

Data: Extracted data

Tools: Simulated access to workstation at DAACs, SCFs, and ECS users

Test Input:

Inputs include internal and remote host connectivity to establish virtual terminal sessions within/outside the DAAC and messages sent via command line.

Test Output:

The expected results of this test include terminal sessions to remote DAAC/SCF hosts and the execution of commands received from the remote service.

Success Criteria:

This test is successful when users are able to connect through remote virtual terminal sessions to DAAC and SCF hosts.

Test Procedures:

Test Case ID: BC010.006	
Test Name: Secure Virtual Terminal	
Test Steps:	Comments:
1. Login to local DAAC host.	
2. Telnet over to a remote DAAC host.	
3. Login to the ECS guest server, as a non-registered user with guest privileges	
4. Verify that the current terminal allows access to a remote DAAC host (directories, files, etc.).	
5. Verify that the means to enhance characteristics of the basic virtual device is by mutual agreement between the two communicating parties.	
6. Verify that the virtual terminal is capable of supporting octet.	
7. Logout of remote DAAC host.	
8. Repeat steps 1 through 4 for the following combinations:	
a. SCF to DAAC b. DAAC to SCF c. SCF to SCF	
9. Verify VT service can connect, login, send command, receive command, and update display	
	C-CSS- 63050 ESN-0010#A (c) EOSD0500#A(d, e, f, g, h, i)

4.11.4.7 Test Case 7: Common Facility GUI Based Sessions (BC010.007)

The purpose of the Common Facility GUI Based Sessions test is to demonstrate the graphical user interface for the various tools provided with the CSS Common Facilities and Services.

Test Configuration:

Hardware: Operational DCE Cell, Host Workstations with X-window capability

Software: Operating system software

Data: External generated data

Tools: Simulated access to workstation at DAACs, ECS users,

Test Input:

Inputs include actions to exercise the CSS common facilities.

Test Output:

The expected results of this test include successful demonstration of CSS Common Facilities and Services.

Success Criteria:

This test case will be successful if all CSS Common Facilities and Services operate through GUIs.

Test Procedures:

Test Case ID: BC010.007	
Test Name: Common Facility GUI Based Sessions	
Test Steps:	Comments:
1. Verify X-applications through the operating system software	C-CSS-63060
2. Verify file access and transfer service, bulletin board service, electronic mail service, and remote virtual service threads	Entire communications infrastructure thread verified ESN-0010#A (a, b, c, d, e, f, g, h, i)

4.11.4.8 Test Case 8: Batch File Transfers (BC010.008)

This test verifies the ability to receive and send files from local and remote machines via FTP in batch mode.

Test Configuration:

Hardware: SunOS 5.4, HPUX9, SGI IRIX5

Software: CsFtFTPRelA, ftpSchedTest.C, kftpSchedTest.C

Data: Internal/External data files

Tools: ftp API, kftp API, Rogue Wave 6.1, testdrivers: ftpSchedTest and kftpSchedTest

Test Input:

ASCII or binary batch files.

Test Output:

Files are transferred via ftp and kftp. FTP/KFTP transfer status logs show files were transferred to their designated locations.

Success Criteria:

The batch files are received successfully on the local and remote machines. Events were logged to the appropriate designated locations.

Test Procedures:

Test Case ID: BC010.008	
Test Name: Batch File Transfers	
Test Steps:	Comments:
1. Login to local DAAC host	
2. Attempt to send a file to another machine by submitting a ftp batch process. **cd** /ecs/formal/CSS/REL_A/DOF/src/FTPSCHED/test/phase3 Execute unit test for batch ftp: **ftpSchedTest_sun5**	For the SUN
3. Verify that the files were sent and received by the host machine at the designated time by checking the contents of the directory structures. **ls -al here***	Data is in 'here directories: here, here1, here2, here3, here5 Notice: 'here4' is an intentional missing directory for error checking
4. Attempt to send a file to another machine by submitting a kftp batch process. **cd** /ecs/formal/CSS/REL_A/DOF/src/FTPSCHED/test/phase3 Execute unit test for batch kftp: **kftpSchedTest_sun5**	For the SUN
5. . Verify that the files were sent and received by the host machine at the designated time by checking the contents of the directory structures. **ls -al here***	Data is in 'here directories: here, here1, here2, here3, here5 Notice: 'here4' is an intentional missing directory for error checking
6. Verify Steps 1-5 on HP and SGI platforms	ESN-0010#A (a, b, c, d, e, f, g, h) ESN-0290#A, ESN-0300#A

4.11.4.9 Test Case 9: Verify TSDIS Communications (BC010.009)

This test verifies the TSDIS communications can successfully transfer data files to ECS over a configurable port utilizing the incoming and delivery gateways. Files are transferred via kftp within the distributed computing environment (DCE) cell configuration. This test must demonstrate the capability of the TSDIS gateway to:

- perform protocol conversion between socket messages and OODCE remote procedure calls, listening at configurable port, and place in network byte order all messages

- utilized kerberos identity through encryption, provide data integrity, and authenticate messages through the DCE/Kerberos service
- route requests received based on initialization information, log communication errors, log authentication requests, and receive information from external entities and DCE servers
- exit the process upon receiving a shutdown message from the internal ECS server (incoming gateway)

Test Configuration:

Hardware: SUN workstation, Gateway server, Ingest server, Data server, Kerberossecurity server

Software: CsGwMessage.cxx, CsGwSdpfMsg.cxx, CsGwGateway.cxx, CsGwIncomingGatewayMain.cxx, CsGwDeliveryGatewayMain.cxx, CsGwIntGatewaymain.cxx, CsGwDlvGatewaymain.cxx, CsGwSdpfMain.cxx

Data: configuration file, keytab file, data files via kftp

Tools: Network monitoring tool: Sniffer, Kerberos 5 beta 5, OODCE 1.0.3, Rogue Wave tools.h++, net.h++ 6.04

Test Input:

Users must change the configuration file to reflect the incoming and delivery gateway executables. A configurable port is set for TSDIS gateway. A message is sent to retrieve data from ECS server. Incoming gateway sends a DAN and the delivery gateway verifies data files against the DAN, and logs to a DDN.

Test Output:

Data files are transferred to ECS via kftp. The data server places the data files in specified locations. The incoming gateway validates with a DAA if no errors found, and the delivery gateway sends a DDA for successfully completed data transfers files.

Success Criteria:

This test will be successful if the external client can connect to a configurable port and verify DAN -> DAA incoming and DDN -> DDA delivery gateways. Also, security considerations must be met.

Test Procedures:

Build Test Case ID: BC010.009	
Test Name: Verify TSDIS Communications	
Test Steps:	Comments:
1. Start the network monitoring tool to monitor communication across the line.	

As a client, login to a DCE Cell, **dce_login** <principal name>; password	
From the monitoring tool, attempt to read information that the server is reading from the client such as: a) client's id b) client's permissions c) client's group d) client's memory capability	
4. Set the port connection through FDDI switch	
Start client session via kftp: **cd /krb5/bin/kftpdir** **kinit** <principal name>; password **klist** **kftp** <host>; username and password	Go the application directory to run "kftp", and gather credential tickets for authentication
Gather authentication response from external process	
Send a DAN to internal process and to the gateway in RPC	
Verify a DAA was returned	
Verify data files were verified against the DAN an logged in the DDN.	
Demonstrate an incomplete data transfer by sending another DAN with corrections inserted	
Perform **kdestroy**	End DCE/kerberos session
Execute these: **dce_login** <principal name>; password **knit** < username> password **klist**	
13. Execute this: **. /** **CsGwDeliveryGatewayMain**	Test driver in window 1(delivery of mesages)
14. Execute this: **. / tisserver** with port number	Test driver in window 2 (TSDIS server: received a DDN and send out a DDA)
15. Execute this: **. / testClient**	Test driver in window 3(simulate Ingest's sending program: sending a DDN and receiving DDA)
16. Incoming gateway test, execute**:** **. / tgclient**	
Execute a failure in the transferring of the file via kftp.	
18. TSDIS sends a new DAN with the corrections	
19. In case of network failure, no retransmissions necessary because TSDIS sends a new DAN with the corrections.	TSDIS does not retransmit in case of fail file transfers
20. To remove credentials: **kdestroy**	
21. Verify files: **cd /krb5/test/** **cat stdout** **cd /krb5/test/** **cat stderr**	Check log for errors

	C-CSS-64040, C-CSS-64080, C-CSS-64090, C-CSS-64010, C-CSS-64020, C-CSS-64030, C-CSS-64060, C-CSS-64070, C-CSS-64100, C-CSS-64110, C-CSS-64120 ESN-0010#A (a, b, c, d, e, f, g, h) ESN-0290#A, ESN-1380#A, ESN-0590#A
22. Disconnect the server connection.	

4.11.4.10 Test Case 10: Verify CSS SDPF Communications (BC010.010)

This test verifies SDPF communications can successfully transfer data files to ECS over a configurable port utilizing a constantly running gateway server that transport messages using the file transfer protocol (ftp). Files are transferred via ftp within the DCE cell configuration. This test must demonstrate the capability of the SDPF gateway to:

- perform protocol conversion between socket messages and OODCE remote procedure calls, listening at configurable port, and place in network byte order all messages
- route requests received based on initialization information, log communication errors, log authentication requests, and receive information from external entities and DCE servers
- exit the process upon receiving a shutdown message from the internal ECS server (incoming gateway)

Test Configuration:

Hardware: SUN workstation, Gateway server, Ingest server, Data server

Software: CsGwMessage.cxx, CsGwSdpfMsg.cxx, CsGwGateway.cxx, CsGwIncomingGatewayMain.cxx, CsGwDeliveryGatewayMain.cxx, CsGwIntGatewaymain.cxx, CsGwDlvGatewaymain.cxx, CsGwSdpfMain.cxx

Data: configuration file, keytab file, data files via ftp

Tools: Network monitoring tool: Sniffer, Rogue Wave tools.h++, net.h++ 6.04

Test Input:

TSDIS makes a socket connection on a configurable port to the the ECS Gateway or RPC connections on a configurable port.

Test Output:

ECS retrieves the data specified via ftp, logs the data to a DDN, awaits a DDA from SDPF.

Success Criteria:

This test will be successful when port connections are configured and messages are transferred successfully to SDPF.

Test Procedures:

Build Test Case ID: BC010.010	
Test Name: Verify CSS SDPF Communications	
Test Steps:	Comments:
1. Start the network monitoring tool to monitor communication across the line.	
2. As a client, login to a DCE Cell: **dce_login** <principal name>; password **klist**	
3. Open a TCP/IP connection on a specified port	
4. A authentication request is performed by SDPF client to the CSS-SDPF Gateway. Verify authentication request through DCE authorization.	
5. Verify configuration file is set for SDPF.	
6. Start the gateway main program with the configuration file name.	
7. Execute: **ingest_recv_DAN**	simulation of internal Ingest server to receive the outgoing DAN message (gateway for listening to a configurable port)
8. Execute: **ingest_send_DDN**	simulation of internal Ingest server to send out the DDN message (DDA is also received)
9. Execute: **sdpfClient**	sdpf client unit test simulation of internal Ingest server to send out the DDN message (requires client with server name, port number, and DAN_file_name)
10. Verify the SDPF gateway will make rpcs to Ingest Server with the messages and receive the response message from the rpc.	
11. Make sure the gateway is always running, status checks: **ps -aceg |grep sdpfmain**	
12. Verify the DDN has the file transfer results.	
13. Logout of the DCE Cell and disconnect the server connection. **Kdestroy**	kills credentials: DCE/kerberos
	C-CSS-64010, C-CSS-64020 C-CSS-64030, C-CSS-64060, C-CSS-64070, C-CSS-64100, C-CSS-64110 C-CSS-64120, ESN-0010#A (a, b, c, d, e, f, h) ESN-0290#A

4.11.4.11 Test Case 11: SDPF Ingest and Archive Test (BC010.011)

This testcase demonstrates the ability to integrate the CSS functionality as it relates to the receiving and ingesting of SDPF data. SDPF creates a connection to an ECS DAAC Gateway

and then sends an authentication request. The ECS DAAC verifies whether or not SDPF is a valid data provider and sends an authentication response. If valid, SDPF then sends a DAN via message passing to the ECS DAAC where it is validated and the ECS DAAC sends a DAA in response. The ECS DAAC then FTPs the data to a Working Storage location that has been allocated and metadata is then extracted. The Advertising BBS Service is called to determine the appropriate Data Server to send an insert request to. The insert request is then submitted for data archiving. The insert is accepted, validated and acknowledged by the Science Data Server. The metadata is validated and the DBMS (inventory) is updated. The science data and metadata are then placed in the archive location. The ECS DAAC then sends a DDN to SDPF to indicate successful archival of the science data and SDPF responds with a DDA which signals the end of the ingest and archive activities related to that specific DAN. All detailed ingest and archive activities are recorded in the Ingest History Log as well as the MSS Event Log.

Test Configuration:

Hardware: Ingest Client, Ingest DBMS Server, Science Data Server, SDPF Client Host, Working Storage and Archive Storage

Software: Ingest Client Interface CSC, Ingest Request Processing CSC, Ingest Working File Collection CSC, Ingest DBMS CSC, Ingest Data Preprocessing CSC, Metadata CSC, Server CSC, SDRV, STGMT

Data: TRMM CERES L0 and Definitive Orbit data

Tools: SDPF Ingest Simulator

Test Input:

Inputs to this test include Ingest Requests (automated electronic network) for CERES L0 and Definitive Orbit data.

Test Output:

Outputs to this test include monitoring of Ingest Requests; screen displays of the following messages: Authentication Response, DAA and DDN; screen displays of all logs; database queries and archive directory listings.

Success Criteria:

This test is deemed successful if the ingest and archive of all data is successful. The ingest activities were successfully monitored. The Authentication Response and DAA messages indicate acceptance and the DDN message indicates a successful archival of data. All logs were updated with detailed information. The database queries verify that the core metadata for all data has been inserted into the DBMS (inventory). The directory listings of the archive verify that all data has been successfully archived.

Test Procedures:

Test Case ID: BC010.011	
Test Name: SDPF Ingest and Archive Test	
Test Steps:	Comments:
1. Login to the Ingest Client workstation. **> rlogin** *<hostname>* **>** *<password>*	Assumption: all necessary processes for the ingest and archive of all data are active.
2. Open an xterm and use this xterm to start three SDPF Client xterms, an Ingest Client xterm and a Data Server xterm. **> xterm -T SDPF_Client_1 &** **> xterm -T SDPF_Client_2 &** **> xterm -T SDPF_Client_3 &** **> xterm -T Ingest_Client &** **> exit**	
3. On all three SDPF Client xterms login to the simulated SDPF Client Host. **> rlogin** *<hostname>* **>** *<password>*	
4. On the first SDPF Client xterm start up the SDPF Ingest simulator user interface.	
5. On the second SDPF Client xterm start up the SDPF Ingest simulator.	
6. On the second SDPF Client xterm send a valid Authentication Request.	
7. On the first SDPF Client xterm send a DAN containing CERES L0 data.	
8. After the SDPF Ingest simulator sends the DDA, on the third SDPF Client xterm display the Authentication Response, DAA and DDN messages and then remove them.	Verify that the Authentication Response disposition states acceptance, the DAA disposition states acceptance and the DDN disposition states successful.
9. On the Ingest Client xterm display the Ingest MSS Event Log.	Verify that the Ingest MSS Event Log contains detailed entries for the ingest of the CERES L0 data.
10. On the Data Server xterm shutdown the Data Server GUI.	
11. Exit the Data Server xterm.	
12. On the Ingest Client xterm shutdown the Ingest GUI Interface.	
13. Exit the Ingest Client xterm.	
14. On the first SDPF Client xterm shutdown the SDPF Ingest Client simulator.	
15. Exit all three SDPF Client xterms.	
16. Logout of the Ingest Client workstation.	
	AM1-0220#A, ESN-0010#A, ESN-0490#A, ESN-0510#A, ESN-0600#A, ESN-0610#A, ESN-1181#A, ESN-1380#A, ESN-1400#A, EOSD2510#A, EOSD2620#A, EOSD2640#A

4.11.4.12 Test Case 12: FTPPull Data from Archive (BC010.012)

This test demonstrates the ability to distribute data to a specified pull location destination using FTP in batch mode. Data is copied to a designated common system location so the user can copy (or "PULL" the data) to the user workspace. The user will receive an E-Mail message providing the necessary path and file naming information and instructions detailing retrieval of the data. Appropriate messages of the receipt of distribution requests are received. Error conditions tested include invalid pull area specified and attempting to access unauthorized data.

Test Configuration:

Hardware: Distribution Management, Working Storage, Network Pull Volume, Data Repository, Tape Archive.

Software: SDSRV, STMGT, DDIST, AMASS, DSS GUI.

Data: Various TRMM Data, including: CER00, CER02, LIS00, LIS15, LIS30, LIS4L,VIRS, TMI, PR, GV, NESDIS, V0, Instrument Characterization, Historical, Inventory Characteristic, Production History, QA Statistics, Scientific Calibration, Spacecraft Historical, and Correlative.

Tools: DDIST Driver.

Test Input:

Inputs to this test include results from searches, as well as electronic distribution requests for data.

Test Output:

Outputs to this test include status messages and retrieved data placed in the designated area.

Success Criteria:

This test is deemed successful if the user successfully retrieves requested data from the pull volume. When errors are encountered, the appropriate messages are displayed.

Procedure:

Test Case ID: BC010.012	
Test Name: "PULL" Data to Specific Destination	
Test Steps: "	Comments:
1. Log on to a GSFC DAAC Workstation.	
2. Start the Science Data Server **> rlogin <machine_name>** **> source dbrc.csh** **> source dssrc.csh** **> ./sdsrv &**	
3. Source the database on a second xterm. **> rlogin <db_machine_name>** **> source .dbrc**	shrink this xterm

4. Invoke the Science Data Server GUI on a third xterm.	
5. Invoke DDIST driver on fourth xterm. **> ./dttest**	
6. Create a valid distribution request and send.	
7. Verify that distribution request is accepted by selecting '**Tracking**' icon and viewing displayed requests.	
8. Retrieve data from the archive via an FTPPull.	
9. Verify the data is retrieved from the archive and placed on Pull Volume by selecting '**Usage**' icon and then viewing data displayed under the '**Pull Area**' tab. Verify the information in the pull area list is correct.	
10. E-mail notification is received stating data's readiness.	
11. Verify that instructions are received with email .	User receives e-mail notification providing instructions to retrieve requested data.
12. Using the instructions received, copy files to home directory for verification.	Data is successfully copied to the user work space.
13. Repeat steps 6-12 several times with different data listed in the Inventory.	
14. Request another distribution of data from the archive via an FTPPull.	
15. Send valid distribution request. Do not retrieve from Pull Area.	
16. Attempt to retrieve data of another user.	
17. Request to distribution data via FTPPull.	
18. Log on as another user.	
19. Attempt to copy the information from the pull area to your workspace.	
20. Verify that user can not access another user's data in pull directory.	
21. Verify that message is displayed denying FTPPull of data.	
22. Using incorrect data in the distribution request, select data granule and retrieve data from the archive via an FTPPull.	Message stating failure to copy files to pull volume is displayed. Working storage, as well as any data copied to the pull volume, is automatically cleaned up.
23. Verify that an error message displayed.	
24. Verify that user is sent a notification of distribution failure.	
25. Verify messages are sent to the appropriate logs.	
26. Terminate all software processes.	
27. Logoff Workstation.	
	ESN-0010#A,ESN-0290#A, ESN-0300#A, ESN-0340#A, ESN-0345#A, ESN0350#A, ESN-0450#A, ESN-0590#A, EOSD5010#A, EOSD2480#A,

4.11.4.13 Test Case 13: FTPPush Data to Archive (BC010.013)

This test demonstrates the ability to "Push" data to a specified destination. An electronic distribution request is submitted for data to be placed in a directory outside the DAAC. The data is copied to the user disk location specified in the request. User will receive e-mail stating that distribution is complete. Several distribution requests are submitted. Error checks will be

performed to attempt to send data to a non-existent directory. User will supply incorrect security information that would not allow the requested data to be placed on the user's system. Messages will be returned stating invalid directory.

Test Configuration:

Hardware: Working Storage HWCI, Network Pull Volume, Data Repository, and external network.

Software: SDSRV, STMGT, DDIST, AMASS, DSS GUI.

Data: Various TRMM Data, including: CER00, CER02, LIS00, LIS15, LIS30, LIS4L,VIRS, TMI, PR, GV, NESDIS, V0, Instrument Characterization, Historical, Inventory Characteristic, Production History, QA Statistics, Scientific Calibration, Spacecraft Historical, and Correlative.

Tools: DDIST Driver.

Test Input:

Inputs to this test include electronic distribution requests using FTPPush.

Test Output:

Outputs to this test include status message displays for data distributed to the designated location.

Success Criteria:

This test is deemed successful if the correct status messages are displayed and the data has been transferred to the correct destination.

Test Procedures:

Test Case ID: BC010.013	
Test Name: "PUSH" Data to Specific Destination	
Test Steps:	Comments:
1. Log on to a GSFC DAAC Workstation.	
2. Start the Science Data Server **> rlogin <machine_name>** **> source dbrc.csh** **> source dssrc.csh** **> ./sdsrv &**	
3. Source the database on a second xterm. **> rlogin <db_machine_name>** **> source .dbrc**	shrink this xterm
4. Invoke the Science Data Server GUI on a third xterm.	
5. Invoke DDIST driver on fourth xterm. **> ./dttest**	
6. Create a valid distribution request specifying FTPPush and execute the request.	

7. Verify that the distribution request is accepted by selecting '**Tracking**' icon and viewing the displayed requests.	
8. Verify that an e-mail message is received saying the data has been pushed to the users system.	
9. Verify that data is pushed to the users system to a directory specified by the user by viewing that directory.	
10. Request another FTPPush to an invalid location.	
11. Verify that a message is displayed stating the distribution failed due to invalid location.	
12. Select data and request an FTPPush to an valid location using invalid security information.	
13. Verify that a message is displayed stating the distribution failed due to invalid security privileges.	
14. Verify that user is sent a notification of distribution failure.	
15. Repeat steps 6-14 several times with different data listed in the Inventory.	
16. Verify messages are sent to the appropriate logs.	
17. Terminate all software processes.	
18. Logoff Workstation.	
	ESN-0010#A,ESN-0290#A, ESN-0300#A, ESN-0340#A, ESN-0345#A, ESN0350#A, ESN-0450#A, EOSD5010#A

4.12 SMC Tests

SMC tests consist of the following builds and threads:

- SMC Ground Events Scheduling Thread (TC036)
- SMC Security Management Thread (TC037)
- SMC Trouble Ticketing Thread (TC038)
- SMC Configuration Management Thread (TC039)
- SMC Build (BC011)

4.12.1 SMC Ground Events Scheduling Thread (TC036)

The Ground Events Planning Service is a complete reuse of the Production Planning Workbench that is part of the Production Planning CSCI provided by the Planning and Data Processing Subsystem (PDPS). The Production Planning Workbench is used to prepare a schedule for the production at a site and forecast the start and completion times of the activities within the schedule.

The Production Planning Workbench provides an interface to submit operations ground events such as production, maintenance, testing, etc. It develops optimum resource utilization plans and schedules based upon approved system configurations and priorities.

The purpose of this thread is to verify the functionality of the SMC Ground Events Scheduling Service which plans and submits ground activity to the resource manager for resources to be scheduled.

Please reference the Ground Events Scheduling Thread (TS051) located in Volume 2 of this document.

4.12.2 SMC Security Management Thread (TC037)

The overall objective of the SMC Security Management Thread is to verify that the system provides for the management of the SMC security mechanisms. These mechanisms are designed to protect and control the access to SMC local resources (data and services), provide security advisory to DAAC sites, and coordinate security recovery from security events. Specifically, both technical and administrative types of security mechanisms for the following items will be verified:

- SMC can perform security tests on a regular basis
- SMC can maintain security logs and metrics
- SMC can receive notification of detected security breaches at the DAAC and other connecting security agencies(such as CERT and NIST),
- SMC can provide security advisory to DAAC sites and coordination of security recovery from security events
- SMC can recover from any security violation/intrusion

4.12.2.1 Test Case 1: SMC Access (TC037.001)

This test case verifies the accessibility to the SMC for authorized users.

Test Configuration:

Hardware: SMC Management workstation

Software: HAL DCE Cell Manager, Acl Entry Editor GUI, Acl Entry GUI

Data: Authorized SMC user account

Tools: SATAN, CRACK, Tripwire, TCP Wrapper

Test Input:

Valid/invalid user accounts to the SMC.

Test Output:

Audit log indicating successful/unsuccessful access to the SMC and a notification to the user of their unauthorized login attempt.

Success Criteria:

This test is successful when users can be allowed/denied access to the SMC.

Test Procedures:

Test Case ID: BC037.001	
Test Name: SMC Access	
Test Steps:	Comments
1. Attempt login to SMC client with invalid userid.	
2. Verify userid is denied and notification is displayed to the user.	
3. Verify login attempt is logged in the MSS audit security log.	
4. Attempt login to SMC client with valid userid and invalid password.	
5. Verify userid is denied and notification is displayed to the user.	
6. Verify login attempt is logged in the MSS audit security log.	
7. Login into SMC client with a valid userid and valid password.	Authorized SMC user login.
8. Verify user is granted access to SMC account.	C-MSS-70330, C-MSS-70340

4.12.2.2 Test Case 2: SMC Security Functions (TC037.002)

This test case verifies the following security function(s) at the SMC:

- perform security tests on a regular and periodic basis
- perform network, communications, and host-based security
- maintain security logs and metrics
- forward security-related information

Test Configuration:

Hardware: SMC specific hardware

Software: HAL DCE Cell Manager, rgy_edit, ACL Entry editor GUI, Acl Entry GUI

Data: None

Tools: HP OpenView, SATAN, CRACK, Tripwire, TCP Wrapper

Test Input:

Simulated SMC security violation test scenario.

Test Output:

SMC maintains security logs and metrics, receives notification of detected security breaches at the DAAC and other connecting security agencies(such as CERT and NIST),

SMC can provide security advisory to DAAC sites and coordination of security recovery from security events

Success Criteria:

This test shall be deemed successful when all security functions can be verified at the SMC for proper implementation..

Test Procedures:

Test Case ID: TC037.002	
Test Name: SMC Security Access	
Test Steps:	Comments
1. Login to SMC workstation.	
2. Simulate a site-wide security violation test.	Record all security configuration settings.
3. Verify SMC security control mechanisms accurately detect security violation attempts into the network or other connecting security agencies(i.e.CERT and NIST).	Use SATAN, Npasswd, CRACK
4. Verify proper violation notifications are received at the SMC.	Use TCP Wrappers C-MSS-70400
5. Verify messages are sent to the appropriate security audit logs.	C-MSS-70410
6. Verify that the SMC notifies all effected sites of security violations and states any actions that need to be taken throughout the network.	C-MSS-70330, C-MSS-70340, C-MSS-70350
7. Analyze and verify security violation is properly resolved within SMC guidelines and procedures.	C-MSS-70420

4.12.2.3 Test Case 3: SMC Security Recovery Procedures (TC037.003)

This test case verifies the ability to recover from any security violation/intrusion at the SMC.

Test Configuration:

Hardware: SMC Resources

Software: HAL DCE Cell Manager, ACL_EDIT, OS ACL, MSS Management Software CI-MCI;

Data: None

Tools: Office Automation tools for compromise recovery, CRACK, Npasswd, SATAN, TCP Wrappers, Tripwire

Test Input:

Operator actions are performed to create a security alert and site compromise. Recovery procedures are then followed and analyzed for effectiveness.

Test Output:

Output includes alarms generated as a result of security alert and site compromise.

Success Criteria:

This test is successful when security recovery instructions and directives are:

- generated with the use of office automation tools
- performed successfully so that the security compromising elements or areas are isolated and/or eliminated from the system.

Test Procedures:

Test Case ID: TC037.003	
Test Name: SMC Security Recovery Procedures	
Test Steps:	Comments
1. Login to SMC site.	
2. Create a security compromise by exceeding a configured security control mechanism.	Alert controls are managed by site policies for security.
3. Verify Office Automation tools provide for the generation of directives for recovery from the security event.	Review documented recovery procedures provided by office automation tools.
4. Verify security compromising elements or areas are isolated and/or eliminated from the system.	C-MSS-70500, C-MSS-70520, C-MSS-70530

4.12.3 SMC Trouble Ticketing Thread (TC038)

For Release A, the SMC operator will access the trouble ticketing service located at each DAAC. Reference the test cases for Trouble Ticketing Thread (TC007).

4.12.4 SMC Configuration Management Thread (TC039)

The purpose of this thread is to verify the functionality of the Systems Logistics Management Service at the SMC, which performs software change management and change request management services. The SMC Configuration Management Service provides the capability to record software life-cycles, dependencies, and maintain modifications. ClearCase is the chosen CM tool for Release A, and Distributed Defect Tracking System (DDTS) is the chosen Change Request Manager tool.

4.12.4.1 Test Case 1: Remove CM Trigger Test (TC039.001)

The shell script, rm_policy.sh, removes triggers from a VOB if user account is vobadm. This test verifies the help option of the shell script, and that the script will inform the user that they must be vobadm to execute this script.

Test Configuration:

Hardware: LSM and SMC workstations

Software: ClearCase, Custom shell script (rm_policy.sh)

Data: Existing VOBs within clearcase (ECS and Science)

Tools: N/A

Test Input:

The user will enter the rm_policy command and script options (-h, for help).

Test Output:

Output includes help information when the user executes rm_policy with the help option, and an error message informing the user they must be vobadm to execute rm_policy successfully.

Success Criteria:

This test is successful when a general user is not allowed to execute this rm_policy script on an existing VOB, and also when it will allow vobadm user to execute the script.

Test Procedures:

Test Case ID: TC039.001	
Test Name: Remove CM Trigger Test	
Test Steps:	Comments:
1. Log into workstation as normal user and set a Clearcase view.	User account is used to test error condition.
2. Execute remove trigger script - **rm_policy.sh**.	The help information for the script will be displayed to the user.
3. Change directory to existing VOB structure, either ECS or Science.	Only 2 VOBs will exist in the Clearcase structure. rm_policy.sh will only execute on an ECS or Science VOB.
4. Execute remove trigger script - **rm_policy.sh**.	An error occurs - **You must be VOBADM to run this script.**
5. Log into workstation as VOBADM and set a Clearcase view.	
6. Change directory to existing VOB structure, either ECS or Science.	Only 2 VOBs will exist in the Clearcase Structure. rm_policy.sh will only execute on an ECS or Science VOB.

7. Execute remove trigger script - **rm_policy.sh**.	Triggers that enforce the CM policies on a VOB will be deleted.
8. To verify all triggers have been removed, execute command - **cleartool lstype -trtype** within the VOB	This command will list all triggers associated with the VOB. None should be found.

4.12.4.2 Test Case 2: Software Library Life-cycle Test (TC039.002)

This test verifies the configuration management system, Clearcase, creates a version history of each software library and software object file. This test also verifies the configuration management system identifies the source and object files and their versions compiled to create each software library or object file.

Test Configuration:

Hardware: SMC workstation

Software: Clearcase, Custom GUI scripts

Data: Existing object/library/source files in clearcase

Tools: N/A

Test Input:

Inputs to this test case include compiled library and object file identifications and xclearcase commands to implement version history reports and configuration record reports.

Test Output:

Outputs to this test include reports displayed within a window, saved to a file, or spooled to a printer.

Success Criteria:

This test is successful when the user is able to generate version history reports and configuration records for library or object files contained within Clearcase.

Test Procedures:

Test Case ID: TC039.002	
Test Name: Software Library Life-cycle Test	
Test Steps:	Comments:
1. Log into workstation and execute **xclearcase**, and select a view.	Xclearcase displays the View browser, after the selection of a view, the File browser will appear.
2. Set the current working directory to a directory that contains library files (lib*.a) or derived object files (*.o). Select a library file or object file, and click on the “tree” icon from the toolbar.	C-MSS-40530 If the library/object file is checked-in to clearcase, a version tree of that file will be displayed. Otherwise the library/object file will contain a date and time from the last successful build.

3. Using the same library/object file, from the menu bar select the option to display the configuration record of that file. Select all options of the display - To Display, To File, To Printer.	C-MSS-40080 SMC-2510 (partial) All the software dependencies will be displayed. Dependencies include software files, versions, compile information, and platform information.
4. Using the same library/object file selection, select from the menu bar, Report Menu, List History. List the history to all options, Display, File, and Printer. Clearcase will prompt for a history type, enter related information.	Clearcase will display the software library/object event history of the selected file.
5. Exit out of xclearcase.	Exits view, and clearcase, and returns user to unix prompt.

4.12.4.3 Test Case 3: CCB Approved Software Changes (TC039.003)

This test demonstrates that a user must identify change configuration requests or non-conformance reports upon modification of source code contained within the configuration management system, Clearcase. This test also verifies that all updates to related CCRs/NCRs are available for system-wide viewing.

Test Configuration:

Hardware: LSM and SMC workstation

Software: Clearcase, Change Request Manager (DDTS), Clearcase to DDTS Integration Package.

Data: Source files in Clearcase, CCR's and NCR's in DDTs

Tools: N/A

Test Input:

Inputs to this test include clearcase commands to check-out and modify existing source files, CCR or NCR ids, and clearcase commands to check-in source files.

Test Output:

Outputs to this test include history reports from the Change Request Manager tracking CCR and NCR status related to source code modification within clearcase.

Success Criteria:

This test is successful when the Change Request Manager records and tracks modifications to files related to CCB-approved change configuration requests or non-conformance reports. These reports will be viewed from LSM and SMC workstations to verify updates. This test is also successful when the user is unable to modify a file when an incorrect CCR of NCR id is entered.

Test Procedures:

Test Case ID: TC039.003	
Test Name: CCB Approved Software Changes	
Test Steps:	Comments:
1. Log into workstation and execute Change Request Manager (DDTS). Enter a software class NCR and submit.	User enters an NCR within DDTS to test the DDTS to Clearcase integration. NCR entered will be used for clearcase check-outs. Record the defect identifier.
3. Set a clearcase view and change current working directory to source code. Execute a **cleartool checkout <***filename***>**	SMC-2535 (partial) The check-out will prompt the user for a defect identifier.
4. Enter an invalid/non-existent defect identifier.	The check-out will verify the defect identifier with DDTS. Since the defect identifier is bogus, the check-out will fail, and cancel.
5. Using the defect identifier from step 1, check-out file again.	The check-out will verify the defect identifier with DDTS. Since the defect exists, check-out is successful. DDTS records the CM Events within the NCR.
6. Verify CM Events enclosure was attached to NCR submitted in step 1.	CM Enclosure should identify *filename* was checked-out.
7. Modify file, and attempt a **cleartool checkin**. Enter invalid defect identifier.	The check-in will verify the defect identifier with DDTS. Since the defect does not exist, the check-in will fail, and cancel.
8. Check-in file again, this time enter 0 or defect identifier from step 1.	Entering 0 at the prompt, will indicate that the defect was not fixed in the code. Entering the defect identifier will indicate defect was fixed. DDTS records all check-in events in the CM Events enclosure within the NCR.
9. Verify CM Events enclosure contains check-in attempts.	CM Enclosure should identify check-in attempts.
10. Execute DDTS from both the SMC workstation and the LSM workstation. Select and view the NCR entered in step 1.	C-MSS-40770 SMC-2530 (partial) NCR should contain CM Enclosure and

4.12.4.4 Test Case 4: Software Turnover Test (TC039.004)

This test verifies the capability of producing a software turnover package including source code, object files, installation scripts and installation instructions for distribution.

Test Configuration:

Hardware: SMC workstation

Software: Software Change Manager, Clearcase

Data: Source files in Clearcase

Tools: N/A

Test Input:

Inputs to this test include standard unix commands to create the software turnover package.

Test Output:

The output of this test case will consist of a file which will contain all identified deliverables.

Success Criteria:

This test will be successful when a software turnover package is produced, and the unpacking of the turnover package includes all desired software.

Test Procedures:

Test Case ID: TC039.004	
Test Name: Software Turnover Test	
Test Steps:	Comments:
1. Log into workstation and set a ClearCase view.	
2. Set current working directory to parent directory of target.	The turnover package will begin from this point. Any files and sub-directories from this point will be packaged.
3. Execute the unix **tar** command to package information into a file. **tar cvf *<tar-filename>***.	A tar file will be created which contains all files and sub-directories from the current working directory.
4. Move the tar file to a file system outside of ClearCase. Using the unix **tar** command, unpackage the file to verify its contents.	C-MSS-40460 AM1-0220 (partial) Verify tar file contains all desired files and subdirectories.
5. Remove tar file and exit ClearCase.	When test complete, clean up any files created from test.

4.12.5 SMC Build (BC011)

The SMC Build consists of integrating and testing the following threads and builds:

- SMC Ground Events Schedule Thread
- SMC Security Management Thread
- SMC Trouble Ticketing Thread
- SMC Configuration Management Thread
- Management Services 2 (LSM) Build

The purpose of this build is to verify the functionality of the above threads and build upon their integration. Overall ECS functionality will be verified when this build is integrated with the Release A DAAC build. This functionality includes the ability for ECS to perform the following functions:

- EOS Mission planning and scheduling

- EOS Mission operations
- Command and control
- Communications and Networking
- Data Input
- Data Processing
- Data Storage
- Data Distribution
- Information Management
- End-to-End Fault Management
- System Management

4.12.5.1 Test Case 1: SMC Configuration Management (BC011.001)

The purpose of the SMC Integrated CM test is to test the Configuration Management thread upon integration with the Management Services 2 Build.

Test Configuration:

Hardware: SMC specific hardware

Software: SMC SW Library, ClearCase, DDTS, Baseline Manager Custom Code

Data: SMC Configuration Records

Tools: None

Test Input:

Inputs to this test case include SCF-provided configuration data for individual algorithms, ECS configuration controlled items, ECS documents, ECS-developed resources, ECS releases, Software-critical and security-sensitive items lists, scientific algorithms and ECS files, change requests via electronic mail, and impact assessment and change evaluation requests.

Test Output:

Outputs to this test case include reports documenting individual versions of CI location and operational status, records maintaining software and hardware change items, baseline changes, maintenance documentation, identity and change status of documents associated with each version, change requests satisfied by each release, historical status records, name and unique identifiers for the entered ECS CIs and a posted toolkit software file.

Success Criteria:

This test is successful when all of the documented test steps have been successfully completed.

Test Procedures:

Test Case ID: BC011.001	
Test Name: SMC Configuration Management	
Test Steps:	Comments:
1. Enter an algorithm into the CMAS.	
2. Verify that the algorithm information includes the SCF provided data (algorithm development version numbers, identification codes, reference numbers, SCF point of contact's name and organization, associated file names, formats, sizes and descriptions, number of files by category and type)	
3. Enter a configuration controlled item into the CMAS.	
4. Verify that the SMC makes available system-wide a name and unique identifier for the entered ECS configuration-controlled items including all of the appropriate ECS information.	
5. Verify that the ECS controlled items are characterized as either system-wide or site-specific.	
6. Verify that information containing the sites where individual versions of controlled items are located and the operational status of that version at the site are obtainable.	
7. Update a previous ECS hardware or software resource (thus updating the version number).	
8. Verify that SMC maintains records that identify the current and previous versions (ECS hardware and software resources and documents) and makes this information available system wide.	
9. Update a software item, changing the version number of the release at each site (repeat for a hardware item).	
10. Verify that the SMC retains records of this baseline change and distributes the records to each site.	
11. Verify that the SMC retains reports indicating specifications and technical, operations, and maintenance documentation, and the identity and change status of documents associated with each version of ECS hardware and software resources deployed and distributes it to each site.	
12. Deliver a new version of software to the sites, verifying that the SMC maintains and distributes to the sites a report which describes the change requests satisfied by the new versions (repeat this step for new hardware and documentation releases).	
13. Verify that historical status records (latest baseline and changes, baseline history, latest release documentation, etc.) can be easily retrieved from the SMC.	
14. Update a software release that only affects a limited number of sites.	
15. Verify that historical status records of ECS baseline changes to sites affected, and installation schedule and installation status are easily retrievable from the SMC.	
16. Verify that the SMC maintains a software-critical and security-sensitive items list.	
17. Verify that the identity and change status of individual ECS resources deployed to the sites is reported by the SMC.	

18. Verify that the SMC shall assemble unlicensed toolkit software files to be posted to the bulletin board. These files should consist of source code, linkable object code, make files, and installation procedures.	SMC-2320#A, SMC-2500#A, SMC-2505#A, SMC-2510#A, SMC-2515#A, SMC-2520#A, SMC-2530#A, SMC-2535#A, SMC-2540#A, ESN-0690#A

4.12.5.2 Test Case 2: SMC Security Management (BC011.002)

The purpose of the SMC Integrated Security test is to test the Security Management thread upon integration with the Management Services 2 Build.

Test Configuration:

Hardware: SMC specific hardware

Software: DCE Cell Manager, TCP Wrapper, TripWire

Data: SMC Configuration Records

Tools: Automation tools for automatic recovery of SMC security violations/intrusions.

Test Input:

Inputs to this test case include login attempts with valid/invalid user name/password combinations, password change, password reset, security registry entrants, drivers to generate process to process actions, DCE administrator privileges, rgy_edit commands, network traffic to simulate intrusion or denial of access, log inputs from the DAACs, populated system management database, recovery procedures, routing configurations, test packets and simulated network load.

Test Output:

Test outputs include systems management logs showing failed invalid process to process actions, a variety of system logs, and screen outputs showing the success or failure of the maintenance transactions. Also, listings of ACLs indicating changes, alarms in OpenView, reports from both SMC and LSM, and network logs showing the success of the packets to get to their destination and detection of anomalous network events.

Success Criteria:

This test is successful when non-authenticated actions fail and get logged and authenticated actions are allowed, allowed privileges prior to ticket expiration and disallowed privileges upon ticket expiration, valid maintenance transactions processed successfully, documented invalid transactions, valid ACL privileges allowed and invalid privileges disallowed, identification of non compliance items, correctly logged data at both the LSM and SMC, correct identification of network anomalies and in all cases compliance with the requirements.

Test Procedures:

Test Case ID: BC011.002	
Test Name: SMC Security Management	
Test Steps:	Comments:
1. Attempt to login using combinations of valid/invalid user names and passwords.	
2. Change a user's password.	
3. Reset a user's password.	
4. Initialize drivers to invoke both valid and invalid process to process actions.	
5. Upon successful login, verify that user has been granted a ticket.	
6. Set the ticket expiration time for a shorter duration.	
7. Logout and log back in.	
8. Verify that user can execute a binding call.	
9. Once ticket time expires, verify that user can not execute a binding call.	
10. Login as a valid DCE administrator.	
11. Perform add, change and delete transactions on ACL files.	
12. Verify that the ACL has been updated.	
13. Verify that the ACL users are only able to execute the privileges they have been granted.	
14. Create security management events.	
15. Verify that the events have been captured.	
16. Run the security management log analysis program.	
17. Run security compliance test reports for both LSM and SMC and verify that the reports are consistent.	
18. Run all appropriate reports and detection tools.	
19. Run the network traffic generator to load the network.	
20. Verify that the system can detect the increased volume.	SMC-0340#A, IMS-0080#A, SMC-0350#A

4.12.5.3 Test Case 3: SMC Ground Events Scheduling (BC011.003)

The purpose of the SMC Integrated Ground Events test is to test the Ground Events Scheduling thread upon integration with the Management Services 2 Build.

Note: The SMC shall have access to the PDPS Resource Manager workbench which shall give the SMC overall site-wide ground scheduling management capability. All capability required for SMC ground events scheduling shall be satisfied through the demonstration of that capability on the PDPS Resource Manager. The SMC requirements are mapped to PDPS tests.

Test Configuration:

Hardware: SMC specific hardware

Software: Production Planning Workbench

Data: SMC Configuration Records

Tools: None

Test Input:

Resource Manager

Test Output:

Timeline display of resources.

Success Criteria:

A ground event will be added, deleted, and modified by the Planning Workbench at the SMC.

Test Procedures:

Test Case ID: BC011.003	
Test Name: SMC Ground Events Scheduling	
Test Steps:	Comments:
These RBR's shall be demonstrated in PDPS testcase TS051.001.	SMC-0320#A, SMC-1300#A, SMC-1310#A, SMC-1320#A, SMC-1345#A, SMC-1360#A

4.12.5.4 Test Case 4: SMC Trouble Ticketing (BC011.004)

The purpose of the SMC Integrated Trouble Ticketing test is to verify the Trouble Ticketing thread upon integration with the Management Services 2 Build.

Note: The SMC shall have authorized access to Trouble Ticketing functionality and capability at each of the local DAAC sites. There are no specific requirements listed for the SMC.

Test Configuration:

Hardware: Local DAAC

Software: Management Services (Remedy Action Request System), User Interface, Service Requester

Data: Impact of the problem - severity number change Description - 1 line Detail problem descriptionShort description SMC Configuration Records

Test Input:

SMC E-Mail address

SMC Configuration Records

Test Output:

Notification of receipt

Success Criteria:

The test is successful when the Trouble Ticketing Service has forwarded selected trouble tickets to another organization and receipt of delivery has been returned.

4.12.5.5 Test Case 5: SMC Scenario (BC011.005)

The purpose of the SMC scenario test is to verify Release A Management Services functionality. The testing will take place at the EDF in an environment that emulates the planned production SMC.

Test Configuration:

Hardware: HP Server

Software: Management Services (Remedy Action Request System), User Interface, Service Requester

Data: SMC Configuration Records, various combinations of valid/invalid password/ID combinations, fault and performance data,

Tools: None

Test Input:

Operator Keyins

Test Output:

Logs indicating invalid/valid login attempts. Summary performance reports and notifications of performance metrics. Screen outputs notifying the user that they have entered an invalid user name or password. Reports for user registration and accountability audit trails, a new software version and a CCR.

Success Criteria:

This test is successful when all of the above functionality has been verified and/or demonstrated.

Test Procedures:

Test Case ID: BC011.005	
Test Name: SMC Scenario	
Test Steps:	Comments:
1. Inject a fault at the emulated LSM located in the EDF. A variety of faults will be incurred in a variety of ways.	
2. Upon appropriate action performed by the LSM and when the predefined conditions (to cause SMC notification) have been met, verify that the SMC receives appropriate notification of the fault condition from the Fault Management Service at the emulated LSM.	

3. Verify that the SMC notifies the other emulated LSMs indicating that a fault has occurred and ensure a quick resolution so that the fault will not occur at another emulated DAAC.	
4. Produce reports based on the information collected and received from the various Fault Management Application Sevices.	
5. Verify that the site Performance Management Application Service collects and processes performance data local to the site and periodically sends the information to the SMC.	
6. Evaluate the system-wide trends and system-level performance analysis based on the performance management information obtained from the LSMs.	
7. Produce the system wide summary performance reports and notifications of performance metrics exceeding established thresholds.	
8. Define the security directives and guidelines.	
9. Distribute these guidelines to all of the DAACs.	
10. Log into an emulated DAAC (at the EDF) not meeting all of the criteria defined in the security directives and guidelines by the SMC.	
11. Verify that the user gets flagged and that the information is recorded at the local site.	
12. Verify that for a security violation the SMC is notified.	
13. Verify that the SMC coordinates the recovery from detected security breaches at the sites and external systems.	
14. Verify that the SMC notifies the other DAACs of the attempted security breach.	
15. Upon recovery, verify that the SMC may coordinate recovery from security events via electronic mail and the telephone.	
16. Upon completion of the User Registration and Accountability Audit Trail Reports at the sites, verify that the reports are sent to the SMC.	
17. Check a new file into the CM tool.	
18. Verify that the Software Change Manager automatically creates and safeguards a file version whenever a new or changed file is checked into the library.	
19. Verify that the software library contains the master copy of all software that is deployed to the sites. Open a new CCR.	
20. Verify that upon completion of the change and closure of the CCR the new baseline record is sent to the SMC.	SMC-1320#A, SMC-3300#A, SMC-3305#A

4.12.5.6 Test Case 6: CMAS SMC Functionality (BC011.006)

This test case demonstrates additional CMAS functionality available at the SMC.

Test Configuration:

Hardware: HP Server

Software: Management Services (Remedy Action Request System), User Interface, Service Requester

Data: SMC Configuration Records, various combinations of valid/invalid password/ID combinations, fault and performance data,

Tools: None

Test Input:

Inputs to this test case include SCF-provided configuration data for individual algorithms, ECS configuration controlled items, ECS documents, ECS-developed resources, ECS releases, software-critical and security-sensitive items lists, scientific algorithms and ECS files, change requests via electronic mail, and impact assessment and change evaluation requests. Specific steps include the following:

Demonstrate that the CMAS tracks the names and identifiers for the following items deployed at the sites: ECS subsystems, networks, and configured system and network devices such as workstations, servers, and routers; ECS releases and baselines; ECS hardware and software resources designated as configuration items; technical documentation and test materials; scientific algorithms, including software, data and test materials (DAACs only); algorithm processing logic control and calibration coefficients data; algorithm test documentation, including specifications, data files, scripts.

Test Output:

Outputs to this test case include the demonstration that a variety of records are maintained by the CMAS and that the information mentioned above is available in these references.

Success Criteria:

This test case is successful when all of the information above is demonstrated in the appropriate record maintained by the CMAS at the SMC.

Test Procedures:

Test Case ID: BC011.006	
Test Name: CMAS SMC Functionality	
Test Steps:	Comments:
1. Verify that the SMC CMAS maintains and makes available system wide, all sites where individual versions of CIs are located and the operational status of that version at the site.	
2. Demonstrate the ability of the CMAS to make available system-wide, records that identify the current and previous versions of ECS hardware and software resources deployed to the sites.	
3. Verify that the CMAS at the SMC shall maintain records that identify the current and previous versions of documents associated with the deployed ECS resources.	

4. Demonstrate the ability of the CMAS at the SMC to distribute to each site records that identify the baseline changes included in each release of ECS hardware and software deployed at the site.	
5. Demonstrate the ability to distribute to each site, record that identify the specifications and technical, operations, and maintenance documents associated with versions of ECS hardware and software configuration items deployed to the site.	
6. Verify that the CMAS at the SMC distributes records to each of the sites that describes any changes to the baselines.	
7. Verify that the CMAS at the SMC maintains historical records about ECS CIs system wide that include: current version; current version's specifications and technical, operations, and maintenance documentation; specifications and technical documentation history; "level of assembly" representation of components comprising the item's current release configurations; and version history.	
8. Verify that the CMAS at the SMC maintains historical records about ECS system releases that include: latest baseline plus approved changes; baseline history; latest release documentation; "level of assembly" representation of the subsystem and configuration item versions that comprise the release configuration; history of changes, including changes to subordinate units/components; effectively and installation status at operational sites; and release configuration.	
9. Verify that the CMAS at the SMC maintains historical records about ECS baseline changes that include: sites affected; installation dates; and installation status.	
10. Verify that software-critical and security-sensitive items lists are maintained by the SMC CMAS.	
11. Verify that the service makes available system-wide, reports that contained the identity and change status of individual ECS resources deployed to the sites and reports containing the identity of resources comprising ECS baselines and releases.	SMC-1000#A, SMC-1340#A, SMC-2610#A, SMC-2620#A

4.12.5.7 Test Case 7 EMC CM (BC011.007)

This test case demonstrates the functionality of CMAS at the EMC.

Test Configuration:

Hardware: HP Server

Software: Management Services (Remedy Action Request System), User Interface, Service Requester

Data: SMC Configuration Records, various combinations of valid/invalid password/ID combinations, fault and performance data,

Tools: None

Test Input:

Inputs to this test case include SCF-provided configuration data for individual algorithms, ECS configuration controlled items, ECS documents, ECS-developed resource, ECS releases, software-critical and security-sensitive items lists, scientific algorithms and ECS files, change requests via electronic mail, and impact assessment and change evaluation requests.

Test Output:

A successful login to the local CMAS at each site. An output indicating the characteristics of the resources at each site. Messages sent to the EMC by the site CMAS indicating: a new baseline at one of the sites, descriptions of what makes up the baselines, and version and implementation information. Historical status records, traceability records and algorithms and SDPS data files ingested to the subsystem.

Success Criteria:

This test case is successful when all of the functionality associated with the local CMAS has been demonstrated/tested and verified.

Test Procedures:

Test Case ID: BC011.007	
Test Name: EMC CM Functionality	
Test Steps:	Comments:
1. Log onto the CMAS at each site.	
2. Verify that when a new baseline is configured at one of the sites, the site CMAS identifies the new baseline and reports the information to the EMC.	
3. Verify "level of assembly" descriptions of operational baselines at the sites are made available to the EMC.	
4. Verify that each site maintains information identifying versions and implementation status of configuration-controlled items at the sites and makes this information available to the EMC.	
5. Verify that, at each site, the CMAS maintains historical status records (including current version, documentation history, etc.) for ECS and algorithm software at the site.	
6. Verify that the CMAS at the EMC and individual sites shall maintain records to establish traceability among operation baselines and releases, and maintain "level of assembly" descriptions of controlled item components.	

4.12.5.8 Test Case 8: MSS Internal Interfaces (BC011.008)

This test case verifies that all of the expected MSS internal interfaces have been met.

Test Configuration:

Hardware: HP Server

Software: Management Services (Remedy Action Request System), User Interface, Service Requester

Data: SMC Configuration Records, various combinations of valid/invalid password/ID combinations, fault and performance data,

Tools: None

Test Input:

Inputs to this test case include an event, a message, time, file access, bulletin board, electronic mail, history log data, security events, performance events, request network protocol status, request network hardware status, etc. The SDPS and FOS interfaces and data will be simulated, since at the time of testing the actual interfaces will not be available. Specific keyins will be performed to complete the following:

Demonstrate the functionality of MSS to communicate with simulated SDPS and FOS entities and to pass the agreed upon data types as outlined in Tables 5.1-2 and 5.1-3 of the Communications and System Management Segment (CSMS) Requirements Specification for the ECS Project (304-CD-003-002).

Test Output:

Test outputs include the successful interactions with all of the above interfaces.

Success Criteria:

This test is successful when the MSS has successfully communicated with all of the interfaces.

Test Procedures:

Test Case ID: BC011.008	
Test Name: MSS Internal Interfaces	
Test Steps:	Comments:
1. Verify the ability of the MSS interfaces between the LSM at the sites and the EMC at the SMC to communicate.	
2. Send history log summary data, security events, fault events, performance events and registration data to the EMC from the LSM.	
3. Verify that the EMC can receive and decipher the information.	
4. Send data requests, policy directives, software distribution, and registry data to the EMC to the LSM.	
5. Verify that the LSM can receive and decipher the information.	
6. Send security events, fault events, performance events, and registration data to the LSM from the LSM.	
7. Verify that the LSM can receive and decipher the information.	
8. Verify that the CSS API can send event logger, time, and a message to the MSS.	
9. Demonstrate the ability of the CSS server to provide Electronic Mail, file access, bulletin board, virtual terminal, directory naming, and security data to the MSS.	
10. Demonstrate the ability of the MSS to send service requests to the CSS server.	
11. Demonstrate the ability of the MSS to request, from the ISS, network protocol status and network hardware status.	

12. Verify that the ISS receives network protocol diagnostics, network hardware diagnostics, and router table maintenance from the MSS.	
13. Verify that the ISS can send the MSS network protocol status data and network hardware status data.	

Appendix A. Requirements Verification Matrix

This appendix contains the Level 3 and Level 4 Requirement Traceability Matrix which maps ECS Level 3 and Level 4 requirements to the test cases in this document.

Table A-1. Requirements Verification Matrix (1 of 134)

Test Case ID	Requirement ID	Requirement Text
TC001.001	C-ISS-01040	The ISS shall provide connectivity between the LaRC DAAC and NOLAN for the ingest of L0 CERES data.
	C-ISS-02000	The ISS shall provide connection oriented transport services as specified by the TCP protocol referenced in RFC 793.
	C-ISS-02020	The ISS shall provide connection-less transport services as specified by the UDP protocol referenced in RFC 768.
	C-ISS-02030	The ISS shall provide network layer services as specified by the Internet Protocol (IP) suite referenced in RFC 791.
TC001.002	C-ISS-02000	The ISS shall provide connection oriented transport services as specified by the TCP protocol referenced in RFC 793.
	C-ISS-02010	The ISS shall provide the capability to filter packets based on the port/socket of the transport layer protocol.
	C-ISS-02030	The ISS shall provide network layer services as specified by the Internet Protocol (IP) suite referenced in RFC 791.
	C-ISS-02040	The ISS shall provide the capability to filter packets based upon network layer source and/or destination addresses.
	C-HRD-37000	The ISS networks shall support the used of network and transport layer filtering to control access from internal and external interfaces.
TC001.003	C-ISS-01256	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between CSMS components at the EDC DAAC
	C-HRD-36065	The ISS-INHW CI shall reuse the existing V0 DAAC LAN at EDC for Rel. A.
TC001.004	C-ISS-02050	The ISS shall provide ICMP network layer service as specified by RFC 792
	C-ISS-02060	The ISS shall provide network layer services in compliance with one or more of the following protocols as appropriate to the type of the physical network supported a. IP over Ethernet as specified in RFCs, 894, 895, 826(ARP), 903 (RARP) b. IP over FDDI as specified in RFC 1188, 1390 (ARP, RARP) c. IP over HiPPI (REL.B) d. IP over SMDS -N/A

Table A-1. Requirements Verification Matrix (2 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-ISS-02530	The ISS shall provide services based on the Routing Information Protocol (RIP) referenced in RFC 1058 to route network traffic between the source and destination nodes.
TC001.005	C-HRD-39000	The ISS-INHCI DAAC LANs shall provide transparent portability across heterogeneous site LAN architectures.
TC001.006	C-ISS-01000	The ISS shall interoperate with the V0 WAN to provide IR-1 connectivity as specified in DID 220 " Communications Requirements for the ECS Project. "
	C-ISS-01010	The ISS shall provide and interface between the V0 WAN and LaRC and GSFC DAACS for the purpose of IR-1 interface testing.
	C-HRD-36080	The ISS LANs at the Release-A DAAC sites shall be designed in a manner that allows: a. Nodes to be added to any given LAN segment. b. Additional LAN segments to be added to the LAN.
	C-ISS-01080	The ISS shall reuse the V0 WAN in order to provide connectivity between V0 network nodes and ECS (V1) network nodes and provide interoperability between the systems.
	C-ISS-01090	The ISS shall provide local or metro area connectivity between V0 network nodes and ECS (V!) network nodes at GSFC, LaRC DAAC sites in order to provide interoperability between the systems.
	C-ISS-01195	The ISS shall provide for connectivity with EBNet at the following ECS sites: a. GSFC DAAC b. GSFC EOC (ECS) c. LaRC DAAC
	C-HRD-31000	The ISS shall provide LANs at the following Release A sites: a. GSFC DAAC LAN b. GSFC EOC LAN c. EDC DAAC LAN d. LaRC DAAC LAN e. MSFC- DELETED f. GSFC SMC LAN
TC001.007	C-ISS-01195	The ISS shall provide for connectivity with EBNet at the following ECS sites: a. GSFC DAAC b. GSFC EOC (ECS) c. LaRC DAAC

Table A-1. Requirements Verification Matrix (3 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-HRD-31000	The ISS shall provide LANs at the following Release A sites: a. GSFC DAAC LAN b. GSFC EOC LAN c. EDC DAAC LAN d. LaRC DAAC LAN e. MSFC- DELETED f. GSFC SMC LAN
TC001.008	C-HRD-32000	The ISS-INHW CI shall use physical devices and MAC protocols compatible with the following standards: a. IEEE 802.2 (LLC) b. IEEE 802.3 (MAC for Ethernet) c. IEEE 802.6 (MAC for SMDS)- N/A c. ANSI X3T9.5 (FDDI)
TC001.009	C-HRD-34000	The ISS-INHW CI LAN Analysis Equipment shall provide protocol analysis the transport layer for all ISS protocols and interconnection protocols to MAN/WANs.
TC001.010	C-HRD-34010	The LAN Analysis Equipment shall include: Communications line monitors to store and display up to 10,000 bytes of data sent and received over any of the communications lines at rates of 10MB/sec to 100MB/sec, and supporting the protocols used within and interconnecting ECS. Local Area Network analyzers
TC001.011	C-HRD-36000	The EOC LAN loop delay contribution shall not exceed more than 500 msec (goal 250 msec) seconds of the total ECS delay of 2.5 seconds for emergency real-time commands.
	C-HRD-36010	The EOC Operational LAN backbone shall be able to support a peak traffic rate of 24 Mbps to support AM-1 flows from the Ecom interface.
TC001.012	C-HRD-36090	The EOC Operational LAN shall be able to support 230 network devices without redesign.
	C-HRD-36100	The EOC Operational LAN shall be able to support peak data rates of up to 48 Mbps without redesign.
	C-HRD-36070	The ISS LANs at the GSFC, MSFC and LaRC DAAC sites shall be capable of supporting twice the R-A network traffic load estimates without redesign.
TC001.013	C-HRD-39005	The ISS-INHCI DAAC LANs shall enable expansion to GByte networks including the ability to provide increased volume of data distribution and access.
TC001.014	C-ISS-04020	Backups of all router configurations files shall be maintained at the local DAAC and the Network Management Facility (NMF).

Table A-1. Requirements Verification Matrix (4 of 134)

Test Case ID	Requirement ID	Requirement Text
TC001.016	C-HRD-32010	The ISS-INHW CI physical components and services shall have the capability to be monitored via SNMP agents.
TC002.001	C-CSS-21010	The CSS Security service shall not transmit passwords in clear text across networks.
	C-CSS-21000	The CSS Security service shall provide an API to verify the identity of users.
	C-CSS-21020	The CSS Security service shall provide the capability to create/modify/delete user accounts and privileges in the security registry.
	C-CSS-21030	The CSS Security service shall provide the capability to define/modify/delete group information in the security registry.
TC002.002	C-CSS-21040	The CSS Security service shall provide an API to limit the time after which a login context will expire.
	C-CSS-21050	The CSS Security Service shall provide an API to refresh login contexts before they expire.
TC002.003	C-CSS-21005	The CSS Security service shall provide a unique session key for each client session.
	C-CSS-21070	The CSS Security Service shall provide an API to store server keys associated with servers to a disk file.
	C-CSS-21100	The CSS Security service shall provide an API to challenge the client/server to authenticate itself at the following three levels. a._connect level b._request level c._packet level
	C-CSS-21080	The CSS Security Service shall provide an API to retrieve the server keys associated with services from a disk file at startup time to authenticate the service.
	C-CSS-21090	The CSS Security Service shall provide an API to change the identity of an application process through server keys.
	C-CSS-21060	The CSS Security Service shall provide an API to accept server keys associated with services interactively at the startup of a service.
TC002.004	C-CSS-21170	The CSS Security service shall provide an API to maintain the integrity of the data passing between processes by using checksums at the following three levels: a._connect level b._request level c._packet level
	C-CSS-21200	The CSS Security service shall support the Data Encryption Standard (DES) to encrypt and decrypt data.

Table A-1. Requirements Verification Matrix (5 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-CSS-21180	The CSS Security service shall provide an API to encrypt and send the data passing between processes at the following three levels: a._connect level b._request level c._packet level
	C-CSS-21190	The CSS Security service shall provide an API to receive and decrypt the data passing between processes at the following three levels: a._connect level b._request level c._packet level
TC002.013	C-CSS-01210	The CSS DOF Service shall provide API's to set the identity of a given principle to a given process.
TC002.017	C-CSS-01190	The CSS DOF Service shall provide API's to maintain the integrity of the data to be passed between the client and the server.
	C-CSS-01200	The CSS DOF Service shall provide API's to maintain the privacy of the data to be passed between the client and the server by encrypting and decrypting the data.
	C-CSS-01220	The CSS DOF Service shall support the TCP and UDP communication protocols to communicate between the servers and the clients.
	C-CSS-21170	The CSS Security service shall provide an API to maintain the integrity of the data passing between processes by using checksums at the following three levels: connect level request level packet level
	C-CSS-21180	The CSS Security service shall provide an API to encrypt and send the data passing between processes at the following three levels: connect level request level packet level
	C-CSS-21190	The CSS Security service shall provide an API to receive and decrypt the data passing between processes at the following three levels: connect level request level packet level
TC002.018	C-CSS-01190	The CSS DOF Service shall provide API's to maintain the integrity of the data to be passed between the client and the server.
	C-CSS-01200	The CSS DOF Service shall provide API's to maintain the privacy of the data to be passed between the client and the server by encrypting and decrypting the data.

Table A-1. Requirements Verification Matrix (6 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-CSS-01220	The CSS DOF Service shall support the TCP and UDP communication protocols to communicate between the servers and the clients.
	C-CSS-21180	The CSS Security service shall provide an API to encrypt and send the data passing between processes at the following three levels: connect level request level packet level
	C-CSS-21190	The CSS Security service shall provide an API to receive and decrypt the data passing between processes at the following three levels: connect level request level packet level
	C-CSS-21200	The CSS Security service shall support the Data Encryption Standard (DES) to encrypt and decrypt.
TC002.019	C-CSS-01190	The CSS DOF Service shall provide API's to maintain the integrity of the data to be passed between the client and the server.
	C-CSS-01200	The CSS DOF Service shall provide API's to maintain the privacy of the data to be passed between the client and the server by encrypting and decrypting the data.
	C-CSS-01220	The CSS DOF Service shall support the TCP and UDP communication protocols to communicate between the servers and the clients.
	C-CSS-21180	The CSS Security service shall provide an API to encrypt and send the data passing between processes at the following three levels: connect level request level packet level
	C-CSS-21190	The CSS Security service shall provide an API to receive and decrypt the data passing between processes at the following three levels: connect level request level packet level
	C-CSS-21200	The CSS Security service shall support the Data Encryption Standard (DES) to encrypt and decrypt.
TC002.020	C-CSS-01170	The CSS DOF Service shall provide APIs to set/get the authentication service type to be used between the server and the client.
TC002.021	C-CSS-01170	The CSS DOF Service shall provide APIs to set/get the authentication service type to be used between the server and the client.

Table A-1. Requirements Verification Matrix (7 of 134)

Test Case ID	Requirement ID	Requirement Text
TC002.022	C-CSS-01170	The CSS DOF Service shall provide APIs to set/get the authentication service type to be used between the server and the client.
TC002.023	C-CSS-01180	The CSS DOF Service shall provide APIs to set/get the authorization service type to be used between the server and the client.
TC002.024	C-CSS-01180	The CSS DOF Service shall provide APIs to set/get the authorization service type to be used between the server and the client.
TC003.001	C-MSS-36020	The MSS Management Agent Service shall communicate via ECS management protocol with the MSS Monitor/Control Service to respond to requests for managed object MIB attributes.
	C-MSS-36070	The MSS Management Agent Service shall provide an ECS management agent for network devices.
	C-HRD-32010	The ISS-INHW CI physical components and services shall have the capability to be monitored via S?NMP agents.
TC004.001	C-MSS-36090	The MSS Management Agent Service shall provide an extensible ECS management agent for ECS applications
	C-MSS-36010	The MSS Management Agent Service shall retrieve data from ECS managed objects in test or operational mode.
	C-MSS-36060	The MSS Management Agent Service shall provide an ECS management agent that is configurable to include: a._Community to respond to and set attributes b._Agent location & contact person c._Traps to send d._Events to log & log file name
	C-MSS-36040	The MSS Management Agent Service shall communicate via ECS management protocol with the MSS Monitor/Control Service to send ECS management traps/events to the Monitor/Control Service.
	C-MSS-36050	The MSS Management Agent Service shall communicate via ECS management protocol with the MSS Monitor/Control Service to receive ECS management set message from the Monitor/Control Service.
	C-MSS-36020	The MSS Management Agent Service shall communicate via ECS management protocol with the MSS Monitor/Control Service to respond to requests for managed object MIB attributes.
TC004.002	C-MSS-36040	The MSS Management Agent Service shall communicate via ECS management protocol with the MSS Monitor/Control Service to send ECS management traps/events to the Monitor/Control Service.
	C-MSS-36090	The MSS Management Agent Service shall provide an extensible ECS management agent for ECS applications.

Table A-1. Requirements Verification Matrix (8 of 134)

Test Case ID	Requirement ID	Requirement Text
TC004.003	C-MSS-36040	The MSS Management Agent Service shall communicate via ECS management protocol with the MSS Monitor/Control Service to send ECS management traps/events to the Monitor/Control Service.
TC004.004	C-MSS-36040	The MSS Management Agent Service shall communicate via ECS management protocol with the MSS Monitor/Control Service to send ECS management traps/events to the Monitor/Control Service.
TC004.005	C-MSS-36040	The MSS Management Agent Service shall communicate via ECS management protocol with the MSS Monitor/Control Service to send ECS management traps/events to the Monitor/Control Service.
TC004.006	C-MSS-36020	The MSS Management Agent Service shall communicate via ECS management protocol with the MSS Monitor/Control Service to respond to requests for managed object MIB attributes.
	C-MSS-36050	The MSS Management Agent Service shall communicate via ECS management protocol with the MSS Monitor/Control Service to receive ECS management set message from the Monitor/Control Service.
TC004.007	C-MSS-36040	The MSS Management Agent Service shall communicate via ECS management protocol with the MSS Monitor/Control Service to send ECS management traps/events to the Monitor/Control Service.
	C-MSS-36090	The MSS Management Agent Service shall provide an extensible ECS management agent for ECS applications.
TC005.001	C-MSS-18350	The MSS Management Data Access Service shall provide the capability for an application to load log files into the management database at the site.
	C-MSS-18050	The MSS Management Data Access Service's shall utilize CSS Services to access/transfer management data.
	C-MSS-18260	The MSS Management Data Access Service shall have the capability to schedule the transfer and loading log files into the management database at the site.
	C-MSS-18270	The MSS Management Data Access Service shall have the capability to schedule the archiving of log files at the site.
	C-MSS-18280	The MSS Management Data Access Service shall have the capability to schedule the transfer of management data at the sites to the SMC.
	C-MSS-18330	The MSS Management Data Access Service shall provide the capability for an application to append records to a log file.
TC007.001	C-MSS-57510	The Trouble Ticketing Service shall provide the ability to automatically notify the originator of the trouble ticket of changes in status.
TC007.002	C-MSS-57530	The Trouble Ticketing Service shall provide the ability to search historical and current trouble tickets by various criteria including keyword, user id, and trouble ticket ID.

Table A-1. Requirements Verification Matrix (9 of 134)

Test Case ID	Requirement ID	Requirement Text
TC007.003	C-MSS-57540	The Trouble Ticketing Service shall provide the ability to forward trouble tickets from one organization to another to facilitate the escalation of trouble tickets (e.g. from DAAC to SMC).
TC007.004	C-MSS-57550	The Trouble Ticketing Service shall provide the ability to maintain different trouble ticket statuses including: Open, Work-In-Progress, Closed, Archived.
TC007.005	C-MSS-57560	The Trouble Ticketing Service shall provide the ability to search for trouble tickets relating to the same resource (equipment).
TC007.006	C-MSS-57580	The Trouble Ticketing Service shall provide the ability to store the following minimum set of information : unique trouble ticket ID, status, description, associated resources, problem solution, originator, keywords.
TC007.007	C-MSS-57610	The Trouble Ticketing Service shall provide the capability to generate reports from the its data.
	C-MSS-57620	The Trouble Ticketing Service shall allow output of reports to either the screen or printer.
TC007.008	C-MSS-57630	The Trouble Ticketing Service shall provide customization features to allow sites to specify notification and escalation rules.
TC007.009	C-MSS-57500	The Trouble Ticketing Service shall have a graphical user interface to support the entry and editing of trouble tickets.
	C-MSS-57600	The Trouble Ticketing Service shall allow entry of a trouble ticket by any registered user of the system.
TC007.010	C-MSS-57520	The Trouble Ticketing Service shall provide an Application Program Interface which supports integration of entry of trouble tickets by other packages.
TC008.001	C-MSS-40400	The MSS configuration management application service at the sites and the SMC shall maintain software libraries to store files containing versions and platform variants of : _source code _binaries and executables _patches _calibration coefficients and control data _scripts _designs and design specifications _databases _technical documentation (both text and graphics) _test data _test reports _interface specifications _configuration data (IR-1)
	C-MSS-40480	The MSS configuration management application service shall use a checkout/edit/checkin paradigm to govern changing of software library files.

Table A-1. Requirements Verification Matrix (10 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-40490	The MSS configuration management application service shall track each software library file that has been changed as a new version of the original file.
	C-MSS-40510	The MSS configuration management application service shall maintain records of actual changes made to ECS software library files in implementing system enhancement requests.
	C-MSS-40570	The MSS configuration management application service shall maintain an audit trail of all changes made to software library files.
	C-MSS-40990	The MSS configuration management application service shall log the following information for configuration management events: _operation type _userid of initiator _date-time stamp _host name
	C-MSS-40995	The MSS configuration management application service shall generate chronological reports of logged CM events associated with M&O staff-selectable: _time frames _operator types _userids _hosts
TC008.002	C-MSS-40410	The MSS configuration management application service at DAAC shall maintain user definable software configuration status information for each algorithm.
	C-MSS-40420	The MSS configuration management application service at each site shall maintain M&O staff definable software configuration status information for each version of every software library file.
	C-MSS-40470	The MSS configuration management service shall regulate operations on software library files through use of individual and group permissions.
	C-MSS-40530	The MSS configuration management application service shall identify implementation status for each version of every software library file, reflecting the life cycle stage to which it has been promoted.

Table A-1. Requirements Verification Matrix (11 of 134)

Test Case ID	Requirement ID	Requirement Text
TC008.003	C-MSS-40400	The MSS configuration management application service at the sites and the SMC shall maintain software libraries to store files containing versions and platform variants of : _source code _binaries and executables _patches _calibration coefficients and control data _scripts _designs and design specifications _databases _technical documentation (both text and graphics) _test data _test reports _interface specifications _configuration data (IR-1)
	C-MSS-40420	The MSS configuration management application service at each site shall maintain M&O staff definable software configuration status information for each version of every software library file.
	C-MSS-40460	The MSS configuration management application service at the SMC shall assemble unlicensed toolkit software files for posting to the ECS bulletin board. Files consist of : _source code _linkable object code for selected workstation configurations _makefiles that automate installation _installation instructions
	C-MSS-40480	The MSS configuration management application service shall use a checkout/edit/checkin paradigm to govern changing of software library files.
	C-MSS-40490	The MSS configuration management application service shall track each software library file that has been changed as a new version of the original file.
	C-MSS-40510	The MSS configuration management application service shall maintain records of actual changes made to ECS software library files in implementing system enhancement requests.
	C-MSS-40530	The MSS configuration management application service shall identify implementation status for each version of every software library file, reflecting the life cycle stage to which it has been promoted.
	C-MSS-40540	The MSS configuration management application service shall perform builds of baseline systems for ECS platforms and audit the builds such that they can be repeated.
	C-MSS-40570	The MSS configuration management application service shall maintain an audit trail of all changes made to software library files.

Table A-1. Requirements Verification Matrix (12 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-40990	The MSS configuration management application service shall log the following information for configuration management events: _operation type _userid of initiator _date-time stamp _host name
	C-MSS-40995	The MSS configuration management application service shall generate chronological reports of logged CM events associated with M&O staff-selectable: _time frames _operator types _userids _hosts
TC008.004	C-MSS-40550	The MSS configuration management application service shall reconstruct previous versions of software library files.
	C-MSS-40990	The MSS configuration management application service shall log the following information for configuration management events: _operation type _userid of initiator _date-time stamp _host name
	C-MSS-40995	The MSS configuration management application service shall generate chronological reports of logged CM events associated with M&O staff-selectable: _time frames _operator types _userids _hosts
TC008.005	C-MSS-40500	The MSS configuration management application service shall merge versions of software library files and identify version conflicts, if any.
	C-MSS-40560	The MSS configuration management application service shall allow concurrent user access to software library files.
TC009.001	C-MSS-87510	The Physical Configuration Management Service shall provide a graphical interface for adding to and editing the existing floor plan.
	C-MSS-87570	The Physical Configuration Management Service shall provide a graphical interface for changing the location of the system components.
	C-MSS-87560	The Physical Configuration Management Service shall provide a graphical interface for viewing the physical location of system components on the floor plans.
	C-MSS-87640	The Physical Configuration Management Service shall provide the ability to display the status obtained above within the graphical interface.

Table A-1. Requirements Verification Matrix (13 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-87630	The Physical Configuration Management Service shall provide the ability to interface with the ECS Management framework to capture status information on each component.
	C-MSS-87620	The Physical Configuration Management Service shall provide tight integration with the Trouble Ticketing (TT) System including allowing direct access of the TT through the Physical Configuration Management Interface.
TC009.002	C-MSS-87600	The Physical Configuration Management Service shall provide a standard set of reports against this data.
	C-MSS-87610	The Physical Configuration Management Service shall provide the ability to produce custom reports against this data.
TC009.003	C-MSS-87500	The Physical Configuration Management Service shall be capable of importing floor plans from existing files .
	C-MSS-87530	The Physical Configuration Management Service shall be capable of determining and storing the following information regarding physical components: a. physical device identification b. physical device information c. physical device location d. physical device status
	C-MSS-87540	The Physical Configuration Management Service shall have the capability to augment the information obtained from ECS Management framework on each component with additional information.
	C-MSS-87580	The Physical Configuration Management Service shall be capable of maintaining the following information for all of the physical system components: a. Inventory data (name, purchase date, purchase price, installation date, manufacturer, serial number, physical location) b. Network data (network location, protocols) c. Maintenance data (maintenance date)
	C-MSS-87610	The Physical Configuration Management Service shall provide the ability to produce custom reports against this data.
	C-MSS-87600	The Physical Configuration Management Service shall provide a standard set of reports against this data.
	C-MSS-87590	The Physical Configuration Management Service shall be capable of interfacing with the Management Database in order to store and retrieve data.
	C-MSS-87550	The Physical Configuration Management Service shall have the capability to allow the entry and storage of information regarding additional physical components that cannot be discerned through the ECS Management framework.

Table A-1. Requirements Verification Matrix (14 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-87520	The Physical Configuration Management Service shall be capable, through interfacing with the ECS Management framework, of determining and storing information regarding physical components.
TC009.004	C-MSS-87510	The Physical Configuration Management Service shall provide a graphical interface for adding to and editing the existing floor plan.
	C-MSS-87520	The Physical Configuration Management Service shall be capable, through interfacing with the ECS Management framework, of determining and storing information regarding physical components.
	C-MSS-87530	The Physical Configuration Management Service shall be capable of determining and storing the following information regarding physical components: a. physical device identification b. physical device information c. physical device location d. physical device status
	C-MSS-87540	The Physical Configuration Management Service shall have the capability to augment the information obtained from ECS Management framework on each component with additional information.
	C-MSS-87590	The Physical Configuration Management Service shall be capable of interfacing with the Management Database in order to store and retrieve data.
TC010.001	C-CSS-22000	The CSS Message Service shall provide an API for senders to send messages to receivers asynchronously without waiting for the receivers to receive it.
	C-CSS-22090	The CSS Message Service shall provide the capability to locate and send (push model) the messages to receivers.
	C-CSS-22100	The CSS Message Service shall provide a non-blocking API for the receiver to contact the message queue and get (pull model) the message.
	C-CSS-22130	The CSS Message Service shall associate the receiver to a returned value and maintain that the information locally until the sender requests that information.
	C-CSS-22140	The CSS Message Service shall provide an API for the sender of the message to receive return information stored at the message queue.
TC010.002	C-CSS-22000	The CSS Message Service shall provide an API for senders to send messages to receivers asynchronously without waiting for the receivers to receive it.
	C-CSS-22090	The CSS Message Service shall provide the capability to locate and send (push model) the messages to receivers.

Table A-1. Requirements Verification Matrix (15 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-CSS-22100	The CSS Message Service shall provide a non-blocking API for the receiver to contact the message queue and get (pull model) the message.
	C-CSS-22110	The CSS Message Service shall support guaranteed delivery of the message to the receiver.
	C-CSS-22120	The CSS Message Service shall provide an API for the sender of the message to get the acknowledgment information the message service receives from the receivers.
	C-CSS-22130	The CSS Message Service shall associate the receiver to a returned value and maintain that the information locally until the sender requests that information.
	C-CSS-22140	The CSS Message Service shall provide an API for the sender of the message to receive return information stored at the message queue.
TC010.003	C-CSS-22120	The CSS Message Service shall provide an API for the sender of the message to get the acknowledgment information the message service receives from the receivers.
	C-CSS-22130	The CSS Message Service shall associate the receiver to a returned value and maintain that the information locally until the sender requests that information.
TC010.004	C-CSS-22000	The CSS Message Service shall provide an API for senders to send messages to receivers asynchronously without waiting for the receivers to receive it.
	C-CSS-22090	The CSS Message Service shall provide the capability to locate and send (push model) the messages to receivers.
	C-CSS-22100	The CSS Message Service shall provide a non-blocking API for the receiver to contact the message queue and get (pull model) the message.
	C-CSS-22110	The CSS Message Service shall support guaranteed delivery of the message to the receiver.
	C-CSS-22120	The CSS Message Service shall provide an API for the sender of the message to get the acknowledgment information the message service receives from the receivers.
	C-CSS-22130	The CSS Message Service shall associate the receiver to a returned value and maintain that the information locally until the sender requests that information.
	C-CSS-22140	The CSS Message Service shall provide an API for the sender of the message to receive return information stored at the message queue.
TC010.005	C-CSS-22000	The CSS Message Service shall provide an API for senders to send messages to receivers asynchronously without waiting for the receivers to receive it.

Table A-1. Requirements Verification Matrix (16 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-CSS-22040	The CSS Message Service shall provide an API for the sender to designate multiple receivers for asynchronous messages.
	C-CSS-22050	The CSS Message Service shall support multiple queues so different groups of processes can use different message queues.
	C-CSS-22090	The CSS Message Service shall provide the capability to locate and send (push model) the messages to receivers.
	C-CSS-22100	The CSS Message Service shall provide a non-blocking API for the receiver to contact the message queue and get (pull model) the message.
	C-CSS-22110	The CSS Message Service shall support guaranteed delivery of the message to the receiver.
	C-CSS-22120	The CSS Message Service shall provide an API for the sender of the message to get the acknowledgment information the message service receives from the receivers.
	C-CSS-22130	The CSS Message Service shall associate the receiver to a returned value and maintain that the information locally until the sender requests that information.
	C-CSS-22140	The CSS Message Service shall provide an API for the sender of the message to receive return information stored at the message queue.
TC010.006	C-CSS-22000	The CSS Message Service shall provide an API for senders to send messages to receivers asynchronously without waiting for the receivers to receive it.
	C-CSS-22090	The CSS Message Service shall provide the capability to locate and send (push model) the messages to receivers.
	C-CSS-22100	The CSS Message Service shall provide a non-blocking API for the receiver to contact the message queue and get (pull model) the message.
	C-CSS-22110	The CSS Message Service shall support guaranteed delivery of the message to the receiver.
	C-CSS-22120	The CSS Message Service shall provide an API for the sender of the message to get the acknowledgment information the message service receives from the receivers.
	C-CSS-22130	The CSS Message Service shall associate the receiver to a returned value and maintain that the information locally until the sender requests that information.
	C-CSS-22140	The CSS Message Service shall provide an API for the sender of the message to receive return information stored at the message queue.
TC010.007	C-CSS-22000	The CSS Message Service shall provide an API for senders to send messages to receivers asynchronously without waiting for the receivers to receive it.

Table A-1. Requirements Verification Matrix (17 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-CSS-22060	The CSS Message Service shall purge a message from the message queue after a user specified time irrespective of its delivery to the receivers.
	C-CSS-22065	The CSS Message Service shall log event messages to the MSS management agents whenever the message service could not deliver a message to any receiver in the time period set by the sender of the message.
	C-CSS-22070	The CSS Message Service shall provide the capability to locate and send the messages to receivers.
TC010.008	C-CSS-22000	The CSS Message Service shall provide an API for senders to send messages to receivers asynchronously without waiting for the receivers to receive it.
	C-CSS-22040	The CSS Message Service shall provide an API for the sender to designate multiple receivers for asynchronous messages.
	C-CSS-22050	The CSS Message Service shall support multiple queues so different groups of processes can use different message queues.
	C-CSS-22090	The CSS Message Service shall provide the capability to locate and send (push model) the messages to receivers.
	C-CSS-22100	The CSS Message Service shall provide a non-blocking API for the receiver to contact the message queue and get (pull model) the message.
	C-CSS-22110	The CSS Message Service shall support guaranteed delivery of the message to the receiver.
	C-CSS-22120	The CSS Message Service shall provide an API for the sender of the message to get the acknowledgment information the message service receives from the receivers.
	C-CSS-22130	The CSS Message Service shall associate the receiver to a returned value and maintain that the information locally until the sender requests that information.
	C-CSS-22140	The CSS Message Service shall provide an API for the sender of the message to receive return information stored at the message queue.
TC011.001	C-CSS-20000	The CSS Directory service shall provide the basic functionality to save and retrieve information into the local namespace: a._Create/Delete/Get context (key) b._List context. c._Set/Get attributes. d._Create/Delete attributes. e._List attributes. f._Set/Get attribute information.
	C-CSS-20060	The CSS Directory service shall provide a way to denote the relative root of the namespace.

Table A-1. Requirements Verification Matrix (18 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-CSS-20110	The CSS Directory service shall determine which naming service to use from a given context.
TC011.002	C-CSS-20000	The CSS Directory service shall provide the basic functionality to save and retrieve information into the local namespace: a._Create/Delete/Get context (key) b._List context. c._Set/Get attributes. d._Create/Delete attributes. e._List attributes. f._Set/Get attribute information.
	C-CSS-20070	The CSS Directory service client shall maintain local cache to keep recently lookup information from the namespace for more efficient further lookups.
TC011.003	C-CSS-20000	The CSS Directory service shall provide the basic functionality to save and retrieve information into the local namespace: a._Create/Delete/Get context (key) b._List context. c._Set/Get attributes. d._Create/Delete attributes. e._List attributes. f._Set/Get attribute information.
TC011.004	C-CSS-20000	The CSS Directory service shall provide the basic functionality to save and retrieve information into the local namespace: a._Create/Delete/Get context (key) b._List context. c._Set/Get attributes. d._Create/Delete attributes. e._List attributes. f._Set/Get attribute information.
TC011.005	C-CSS-20000	The CSS Directory service shall provide the basic functionality to save and retrieve information into the local namespace: a._Create/Delete/Get context (key) b._List context. c._Set/Get attributes. d._Create/Delete attributes. e._List attributes. f._Set/Get attribute information.
TC011.006	C-CSS-20090	The CSS Directory Service shall define a minimum of 20 users defined attribute types for application users to store and retrieve attribute information.
TC011.007	C-CSS-20020	The CSS Directory Service shall provide a mechanism to periodically update copies of the namespace from the namespace designated as the master.

Table A-1. Requirements Verification Matrix (19 of 134)

Test Case ID	Requirement ID	Requirement Text
TC011.008	C-CSS-20020	The CSS Directory Service shall provide a mechanism to periodically update copies of the namespace from the namespace designated as the master.
	C-CSS-20025	The updating of the namespace shall be done: automatically manually by the administrator
TC011.009	C-CSS-20000	The CSS Directory service shall provide the basic functionality to save and retrieve information into the local namespace: a._Create/Delete/Get context (key) b._List context. c._Set/Get attributes. d._Create/Delete attributes. e._List attributes. f._Set/Get attribute information.
TC011.010	C-CSS-20000	The CSS Directory service shall provide the basic functionality to save and retrieve information into the local namespace: a._Create/Delete/Get context (key) b._List context. c._Set/Get attributes. d._Create/Delete attributes. e._List attributes. f._Set/Get attribute information.
BC003.001	EOSD2480#A	ECS elements shall require unique sessions when security controlled data are being manipulated.
	EOSD2640#A	ECS elements shall relinquish a connection between the element and a user when the user has not been active for a configurable period of time.
	ESN-0600#A	The ESN Directory service shall include services and supporting mechanisms to authenticate the credentials of a user for the purpose of granting access rights and authorizing requested operations.
	ESN-1380#A	The ESN shall provide countermeasures for the following security threats related to data communications: a. modification of data (i.e., manipulation) while in transit over the network b. disclosure of authentication information c. degradation in network or processing resource performance through denial of service attack d. Impersonation of authentication credentials or authorization privileges.
BC003.003	ESN-0490#A	The ESN shall provide a name-to-attribute mapping Directory Service at a minimum.

Table A-1. Requirements Verification Matrix (20 of 134)

Test Case ID	Requirement ID	Requirement Text
	ESN-0510#A	The directory function shall be able to respond to requests for information concerning named objects, either physical or logical, so as to support communications with those objects.
	ESN-0610#A	The ESN shall include multiple Directory Service Agents (DSAs) which shall be collectively responsible for holding or retrieving all directory information which is needed by ECS.
TC012.001	C-MSS-75020	The MSS Accountability Management Service shall create a new user account whenever a new record is added to the user profile database.
TC012.002	C-MSS-75000	The MSS accountability management service shall provide the capability to maintain a user profile database that stores the following information for each registered user: a. Name b. User ID c. Mailing address d.Telephone number e E-mail address f. Organization (optional)
TC013.001	C-CSS-60500	The CSS File Access Service shall provide functionality for interactive and non-interactive transfer of files (send and receive) between two host systems.
	C-CSS-60510	The CSS File Access Service shall be capable of transferring ASCII and binary files.
	C-CSS-60520	The CSS File Access Service shall support the File Transfer Protocol (FTP).
	C-CSS-60600	The CSS File Access Service shall provide connection oriented operation for file transfers.
	C-CSS-60610	The CSS File Access Service shall allow selection of the file type (ASCII or binary).
	C-CSS-60920	The CSS File Access Service shall accept authentication information for file transfers.
TC013.002	C-CSS-60800	The CSS File Access Service shall provide an option for scheduling file transfers in a batch mode.
	C-CSS-60810	The CSS File Access Service shall log results of the non-interactive operations to operator specified log files.
	C-CSS-60820	The CSS File Access Service shall provide an option to send alarms and generate events if a scheduled operation fails.
TC013.003	C-CSS-60620	The CSS File Access Service shall support proxy mode of operation which enables transfer of files between two remote hosts.
	C-CSS-60630	The CSS File Access Service shall provide capability to list remote files
	C-CSS-60640	The CSS File Access Service shall support wildcards in files on the remote host.

Table A-1. Requirements Verification Matrix (21 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-CSS-60650	The CSS File Access Service shall support anonymous FTP which allows read access to all users.
TC013.004	C-CSS-60900	The CSS File Access Service shall provide an API which allows applications to transfer files.
TC014.001	C-CSS-64000	The CSS Gateway shall perform a protocol conversion between socket messages and OODCE rpcs.
	C-CSS-64010	The CSS Gateway shall route requests received from external providers based on message type and initialization information.
	C-CSS-64020	The CSS Gateway shall log all communication errors, however, it will not check the validity of the contents of any of the messages between the external providers and the DCE servers.
	C-CSS-64030	The CSS Gateway shall log all authentication requests.
	C-CSS-64040	The CSS Gateway shall authenticate , using the DCE/Kerberos Authentication service, in the behalf of external providers.
	C-CSS-64060	The CSS Gateway shall reject requests from a provider unless a valid authentication request is received.
	C-CSS-64070	The CSS Gateway shall be capable of listening at a configurable port number.
	C-CSS-64080	The CSS Gateway shall be capable of obtaining a well known Kerberos identity with which to encrypt the password if needed.
	C-CSS-64090	The CSS Gateway shall be capable of encrypting or providing data integrity as desired by the client.
	C-CSS-64100	The CSS Gateway shall place in network byte order all messages that it, itself, composes for sending over the socket.
	C-CSS-64110	The CSS Gateway shall be capable of receiving control information from both the external entities and DCE servers.
	C-CSS-64120	The Incoming CSS Gateway shall provide the capability to exit from the process upon receiving a shutdown control message from the internal ECS servers.
TC014.002	C-CSS-64000	The CSS Gateway shall perform a protocol conversion between socket messages and OODCE rpcs.
	C-CSS-64010	The CSS Gateway shall route requests received from external providers based on message type and initialization information.
	C-CSS-64020	The CSS Gateway shall log all communication errors, however, it will not check the validity of the contents of any of the messages between the external providers and the DCE servers.
	C-CSS-64030	The CSS Gateway shall log all authentication requests.
	C-CSS-64040	The CSS Gateway shall authenticate, using the DCE/Kerberos Authentication service, in the behalf of external providers.
	C-CSS-64060	The CSS Gateway shall reject requests from a provider unless a valid authentication request is received.

Table A-1. Requirements Verification Matrix (22 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-CSS-64070	The CSS Gateway shall be capable of listening at a configurable port number.
	C-CSS-64080	The CSS Gateway shall be capable of obtaining a well known Kerberos identity with which to encrypt the password if needed.
	C-CSS-64090	The CSS Gateway shall be capable of encrypting or providing data integrity as desired by the client.
	C-CSS-64100	The CSS Gateway shall place in network byte order all messages that it, itself, composes for sending over the socket.
	C-CSS-64110	The CSS Gateway shall be capable of receiving control information from both the external entities and DCE servers.
	C-CSS-64120	The Incoming CSS Gateway shall provide the capability to exit from the process upon receiving a shutdown control message from the internal ECS servers.
TC015.001	C-CSS-21120	The CSS Security service shall provide an API to check the authorization principle privileges of principles to access/control services resources.
	C-CSS-21220	The CSS Security service shall provide a name based authorization capability.
TC015.002	C-CSS-21160	The CSS Security service shall provide the following APIs to MSS security management applications to retrieve/modify the access control lists associated with the ECS services/resources. a._to identify the permissions available to a principal b._to identify all the ACL managers protecting an object c._to get the printable representation of the permissions d._to locate the server with the writable copy of the ACL e._to read an ACL f._to write an ACL g._to test if the calling principal has some permissions h._to test if another principal has some permissions.
TC015.003	C-CSS-21160	The CSS Security service shall provide the following APIs to MSS security management applications to retrieve/modify the access control lists associated with the ECS services/resources. a._to identify the permissions available to a principal b._to identify all the ACL managers protecting an object c._to get the printable representation of the permissions d._to locate the server with the writable copy of the ACL e._to read an ACL f._to write an ACL g._to test if the calling principal has some permissions h._to test if another principal has some permissions.

Table A-1. Requirements Verification Matrix (23 of 134)

Test Case ID	Requirement ID	Requirement Text
TC015.004	C-CSS-21160	The CSS Security service shall provide the following APIs to MSS security management applications to retrieve/modify the access control lists associated with the ECS services/resources. a._to identify the permissions available to a principal b._to identify all the ACL managers protecting an object c._to get the printable representation of the permissions d._to locate the server with the writable copy of the ACL e._to read an ACL f._to write an ACL g._to test if the calling principal has some permissions h._to test if another principal has some permissions.
TC015.005	C-CSS-21160	The CSS Security service shall provide the following APIs to MSS security management applications to retrieve/modify the access control lists associated with the ECS services/resources. a._to identify the permissions available to a principal b._to identify all the ACL managers protecting an object c._to get the printable representation of the permissions d._to locate the server with the writable copy of the ACL e._to read an ACL f._to write an ACL g._to test if the calling principal has some permissions h._to test if another principal has some permissions.
TC015.006	C-CSS-21160	The CSS Security service shall provide the following APIs to MSS security management applications to retrieve/modify the access control lists associated with the ECS services/resources. a._to identify the permissions available to a principal b._to identify all the ACL managers protecting an object c._to get the printable representation of the permissions d._to locate the server with the writable copy of the ACL e._to read an ACL f._to write an ACL g._to test if the calling principal has some permissions h._to test if another principal has some permissions.
TC015.008	C-CSS-21110	The CSS Security service shall authenticate the principle before checking whether the principle is authorized to access a service/resources.
TC015.009	C-CSS-21105	The CSS will notify MSS upon a failure or success of each authentication request.
	C-CSS-21120	The CSS Security service shall provide an API to check the authorization privileges of principles to access/control services/resources.
TC015.010	C-CSS-21130	The CSS Security service shall provide an API to define the permission schema associated with the server/resource.

Table A-1. Requirements Verification Matrix (24 of 134)

Test Case ID	Requirement ID	Requirement Text
TC015.011	C-CSS-21140	The CSS Security service shall provide an API to create and maintain the ACL's associated with the server/resource in a database.
TC015.012	C-CSS-21140	The CSS Security service shall provide an API to create and maintain the ACL's associated with the server/resource in a database.
TC015.013	C-CSS-21150	The CSS Security service shall provide an API to save/retrieve the ACL database onto persistent store.
TC015.014	C-CSS-21210	The CSS Security service shall provide the capability to log audit information into security logs whenever authentication and authorization services are used. The audit information will contain the following: a._Date and time of the event b._User name c._Type of event d._Success or failure of the event e._Origin of the request
TC015.015	C-CSS-21160	The CSS Security service shall provide the following APIs to MSS security management applications to retrieve/modify the access control lists associated with the ECS services/resources. a._to identify the permissions available to a principal b._to identify all the ACL managers protecting an object c._to get the printable representation of the permissions d._to locate the server with the writable copy of the ACL e._to read an ACL f._to write an ACL g._to test if the calling principal has some permissions h._to test if another principal has some permissions.
TC015.016	C-CSS-21160	The CSS Security service shall provide the following APIs to MSS security management applications to retrieve/modify the access control lists associated with the ECS services/resources. a._to identify the permissions available to a principal b._to identify all the ACL managers protecting an object c._to get the printable representation of the permissions d._to locate the server with the writable copy of the ACL e._to read an ACL f._to write an ACL g._to test if the calling principal has some permissions h._to test if another principal has some permissions.

Table A-1. Requirements Verification Matrix (25 of 134)

Test Case ID	Requirement ID	Requirement Text
TC015.017	C-CSS-21160	The CSS Security service shall provide the following APIs to MSS security management applications to retrieve/modify the access control lists associated with the ECS services/resources. a._to identify the permissions available to a principal b._to identify all the ACL managers protecting an object c._to get the printable representation of the permissions d._to locate the server with the writable copy of the ACL e._to read an ACL f._to write an ACL g._to test if the calling principal has some permissions h._to test if another principal has some permissions.
TC016.001	C-CSS-25020	The CSS Time Service shall be used to obtain timestamps that are based on Coordinated Universal Time (UTC).
	C-CSS-25030	The CSS Time Service shall provide an API to retrieve timestamp information.
	C-CSS-25060	The CSS Time Service shall provide an API for converting between UTC time and local time.
	C-CSS-25110	The CSS Time Service shall utilize a UTC based time provider.
TC016.002	C-CSS-25020	The CSS Time Service shall be used to obtain timestamps that are based on Coordinated Universal Time (UTC).
	C-CSS-25030	The CSS Time Service shall provide an API to retrieve timestamp information.
	C-CSS-25040	The CSS Time Service shall provide an API for converting between binary timestamps that use different time structures.
	C-CSS-25050	The CSS Time Service shall provide an API for converting between binary timestamps and ASCII representations.
	C-CSS-25070	The CSS Time Service shall provide an API for manipulating binary timestamps.
	C-CSS-25080	The CSS Time Service shall provide an API for comparing two binary time values.
	C-CSS-25070	The CSS Time Service shall provide an API for manipulating binary timestamps.
	C-CSS-25090	The CSS Time Service shall provide an API for calculating binary time values.
	C-CSS-25100	The CSS Time Service shall provide an API for obtaining time zone information.
	C-CSS-25110	The CSS Time Service shall utilize a UTC based time provider.
TC016.003	C-CSS-25020	The CSS Time Service shall be used to obtain timestamps that are based on Coordinated Universal Time (UTC).
	C-CSS-25030	The CSS Time Service shall provide an API to retrieve timestamp information.
	C-CSS-25110	The CSS Time Service shall utilize a UTC based time provider.

Table A-1. Requirements Verification Matrix (26 of 134)

Test Case ID	Requirement ID	Requirement Text
TC017.001	C-MSS-36090	The MSS Management Agent Service shall provide an extensible ECS management agent for ECS applications.
TC017.002	C-MSS-36090	The MSS Management Agent Service shall provide an extensible ECS management agent for ECS applications.
TC017.003	C-MSS-36090	The MSS Management Agent Service shall provide an extensible ECS management agent for ECS applications.
TC017.004	C-MSS-36100	The MSS Management Agent Service shall provide proxy agents for ECS network devices and applications that cannot be managed via SNMP.
TC017.005	C-MSS-36100	The MSS Management Agent Service shall provide proxy agents for ECS network devices and applications that cannot be managed via SNMP.
TC017.006	C-MSS-36090	The MSS Management Agent Service shall provide an extensible ECS management agent for ECS applications.
TC017.007	C-MSS-36090	The MSS Management Agent Service shall provide an extensible ECS management agent for ECS applications.
TC017.008	C-MSS-36090	The MSS Management Agent Service shall provide an extensible ECS management agent for ECS applications.
TC017.009	C-MSS-36090	The MSS Management Agent Service shall provide an extensible ECS management agent for ECS applications.
TC018.001	C-MSS-20010	The MSS Discovery Service shall discover new instances of managed objects.
	C-MSS-20020	The MSS Discovery Service shall detect missing occurrences of managed objects.
	C-MSS-20030	The MSS Discovery Service shall report missing occurrences of managed objects.
	C-MSS-20040	The MSS Discovery Service shall update the object database after the Discovery Service receives a request to register/unregister a managed object.
	C-MSS-36040	The MSS Management Agent Service shall communicate via ECS management protocol with the MSS Monitor/Control Service to send ECS management traps/events to the Monitor/Control Service.
	C-MSS-36042	The MSS Management Agent Service shall send ECS management traps/events to the management server using a reliable notification mechanism.
	C-MSS-36045	The MSS Management Agent Service shall send ECS management traps/events to the management server using a secure notification mechanism.

Table A-1. Requirements Verification Matrix (27 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-36060	The MSS Management Agent Service shall provide an ECS management agent that is configurable to include: Community to respond to and set attributes Agent location & contact person Traps to send Events to log & log file name
TC018.002	C-MSS-20020	The MSS Discovery Service shall detect missing occurrences of managed objects.
	C-MSS-20030	The MSS Discovery Service shall report missing occurrences of managed objects.
	C-MSS-36040	The MSS Management Agent Service shall communicate via ECS management protocol with the MSS Monitor/Control Service to send ECS management traps/events to the Monitor/Control Service.
	C-MSS-36060	The MSS Management Agent Service shall provide an ECS management agent that is configurable to include: Community to respond to and set attributes Agent location & contact person Traps to send Events to log & log file name
TC0019.001	C-MSS-36010	The MSS Management Agent Service shall retrieve data from ECS managed objects in test or operational mode.
	C-MSS-36020	The MSS Management Agent Service shall communicate via ECS management protocol with the MSS Monitor/Control Service to respond to requests for managed object MIB attributes
	C-MSS-36060	The MSS Management Agent Service shall provide an ECS management agent that is configurable to include: a._Community to respond to and set attributes b._Agent location & contact person c._Traps to send d._Events to log & log file name
	C-MSS-36070	The MSS Management Agent Service shall provide an ECS management agent for network devices
	C-MSS-36080	The MSS Management Agent Service shall provide an extensible ECS management agent for ECS Host systems.
	C-MSS-36090	The MSS Management Agent Service shall provide an extensible ECS management agent for ECS applications.
	C-MSS-36100	The MSS Management Agent Service shall provide proxy agents for ECS network devices and applications that cannot be managed via SNMP.
	C-MSS-36110	The MSS Management Agent Service shall provide an ECS domain manager agent to coordinate and communicate with multiple ECS management agents.

Table A-1. Requirements Verification Matrix (28 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-HRD-32010	The ISS-INHW CI physical components and services shall have the capability to be monitored via SNMP agents.
TC019.002	C-MSS-36020	The MSS Management Agent Service shall communicate via ECS management protocol with the MSS Monitor/Control Service to respond to requests for managed object MIB attributes
	C-MSS-36050	The MSS Management Agent Service shall communicate via ECS management protocol with the MSS Monitor/Control Service to receive ECS management set message from the Monitor/Control Service.
	C-MSS-36052	The MSS Management Agent Service shall receive ECS management set messages from the management server using a reliable mechanism.
	C-MSS-36055	The MSS Management Agent Service shall receive ECS management set messages from the management server using a secure mechanism.
	C-MSS-36060	The MSS Management Agent Service shall provide an ECS management agent that is configurable to include: a._Community to respond to and set attributes b._Agent location & contact person c._Traps to send d._Events to log & log file name
	C-MSS-36090	The MSS Management Agent Service shall provide an extensible ECS management agent for ECS applications.
TC020.001	C-MSS-66000	The MSS performance management application service shall be capable of monitoring the performance of the following ECS components a._network components _1. routers _2. links _3. bridges _4. gateway
	C-MSS-66050	The MSS performance management application service shall be capable of requesting performance data from each individual managed object: a._at configurable intervals b._on demand.
	C-MSS-66060	The MSS performance management application service shall be capable of receiving requested performance data from ECS components.
	C-MSS-66070	The MSS Performance Management Application Service shall be capable of receiving unrequested performance data from ECS managed objects.

Table A-1. Requirements Verification Matrix (29 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-66080	The MSS performance management application service shall be capable of retrieving the following data for all network component interfaces: a._operational status b._type c._speed d._octets in/out e._packets in/out f._discards in/out g._errors in/out
	C-MSS-66120	The MSS performance management application service shall be capable of determining the operational state of all network components, hosts, and peripherals to be: a._on-line b._off-line c._in test mode
	C-MSS-66130	The MSS performance management application service shall be capable of receiving operational state change notifications from network components, hosts, applications, and peripherals.
	C-MSS-68000	The MSS performance management application service shall be capable of graphically displaying the operational state of managed objects through the MUI service.
	C-MSS-68020	The MSS performance management application service shall be capable of printing M&O staff-selected performance statistics.
TC020.002	C-MSS-66100	The MSS performance management application service shall be capable of retrieving the following data for all hosts: a._total CPU utilization b._memory utilization c._physical disk i/o's d._disk storage size e._disk storage used f._number of active processes g._length of run queue h._network i/o's (packets) i._network errors
	C-MSS-66305	The MSS performance management application service shall be capable of collecting the following performance data for all ECS-managed processes: start time stop time CPU utilization memory utilization disk reads disk writes

Table A-1. Requirements Verification Matrix (30 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-66310	The MSS performance management application service shall be capable of retrieving the following science algorithm performance data via the Management Data Access Service: a._algorithm name b._algorithm version c._start time d._stop time e._CPU utilization f._memory utilization g._disk reads h._disk writes
TC020.003	C-MSS-66010	The MSS performance management application service shall be capable of monitoring ECS component protocol stack performance parameters defined in IETF RFC 1213.
	C-MSS-66020	The MSS Performance Management Application Service shall be capable of monitoring ethernet-like device performance parameters as specified in IETF RFC 1623.
	C-MSS-66030	The MSS performance management application service shall be capable of receiving managed object definitions for each managed object.
	C-MSS-66090	The MSS Performance Management Application Service shall have the capability to collect the following performance information about communication protocol stacks on managed devices: _a._number of transport layer messages received with errors _b._number of transport layer messages requiring retransmission _c._number of transport layer messages received that could not be delivered _d._number of jetwork layer messages received with errors _e._number of network layer messages received that could not be delivered _f._number of network layer messages that were discarded_
TC020.004	C-MSS-66040	The MSS performance management application service shall be capable of specifying which available performance metrics are to be gathered from each individual managed object.
	C-MSS-66170	The MSS performance management application service shall log ECS performance data pertaining to ECS network components and operating system resources.
	C-MSS-66190	The MSS performance management application service shall provide a configurable number of thresholds for each performance metric.
	C-MSS-66200	The MSS EMC performance management application service shall be capable of creating a list of suggested initial threshold values for each performance metric.

Table A-1. Requirements Verification Matrix (31 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-66210	The MSS EMC performance management application service shall be capable of sending a list of suggested initial thresholds for each performance metric to the MSS site performance management application service.
	C-MSS-66220	The MSS site performance management application service shall be capable of receiving a list of suggested initial thresholds for each performance metric from the MSS EMC performance management application service.
	C-MSS-66230	The MSS performance management application service shall allow each performance metric threshold to be configurable.
	C-MSS-66240	The MSS performance management application service shall be capable of evaluating each performance metric against defined thresholds.
	C-MSS-66250	The MSS performance management application service shall record an event in the local History Log whenever a threshold is crossed.
	C-MSS-67000	The MSS performance management application service shall be capable of extracting values of performance metrics gathered for a specified managed objects over a configurable period of time from the Management Database.
TC020.005	C-MSS-66140	The MSS EMC Performance Management Application Service shall have the capability to request performance data from: a._Site performance management applications b._Other external systems as defined in Section 5.1.
	C-MSS-66150	The MSS EMC Performance Management Application Service shall be capable of receiving performance data from: a._Site performance management applications b._Other external systems as defined in Section 5.1.
	C-MSS-66160	The MSS EMC Performance Management Application Service shall be capable of receiving summarized performance data from: a._Site performance management applications b._Other external systems as defined in Section 5.1.
	C-MSS-66180	The MSS performance management application service shall have the capability to generate the following types of statistics for a configurable period of time for performance data stored in the Management Database: a._average b._median c._maximum d._minimum e._ratios f._rates g._standard deviations.

Table A-1. Requirements Verification Matrix (32 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-66260	The MSS performance management application service shall provide queries that generate performance statistics from performance data stored in the Management Database.
	C-MSS-66270	The MSS performance management application service shall store generated performance statistics.
	C-MSS-66280	The MSS site performance management application service shall be capable of extracting summarized site status information from logged performance data.
	C-MSS-66290	The MSS site performance management application service shall be capable of sending summarized status information for that site to the MSS EMC performance management application service.
	C-MSS-66300	The MSS EMC performance management application service shall log received summarized site status.
TC020.006	C-MSS-67010	The MSS performance management application service shall be capable of generating a graph of the extracted performance metric values.
TC020.007	C-MSS-68090	The MSS Performance Management Application Service shall have the capability to generate reports from collected management data.
	C-MSS-68100	The MSS Performance Management Application Service shall have the capability to redirect reports to: a._console b._disk file c._printer
	C-MSS-69000	The MSS performance management application service shall maintain operational benchmark test procedures.
	C-MSS-69010	The MSS performance management application service shall receive and maintain operational benchmark test results.
	C-MSS-69020	The MSS performance management application service shall be capable of performing operational benchmark tests.
	C-MSS-69030	The MSS performance management application service shall be capable of providing results of benchmark tests and results of predefined tests to the M&O staff for validation.
TC020.008	C-MSS-68030	The MSS performance management application service shall be capable of receiving system resource utilization information requests from the SDPS Data Processing subsystem via the Management Agent Service.
	C-MSS-68040	The MSS performance management application service shall be capable of providing the following current system resource utilization information to the SDPS Data Processing subsystem via Management Agent Service: a._CPU utilization b._memory utilization c._disk i/o's (per second)

Table A-1. Requirements Verification Matrix (33 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-68060	The MSS performance management application service shall be capable of providing the following current resource utilization information to the SDPS Data Server subsystem via the Management Agent Service: a._CPU utilization b._memory utilization c._disk I/O's (per second)
	C-MSS-68080	The MSS performance management application service shall be capable of providing the following current resource utilization information to the SDPS Client subsystem via the Management Agent Service. a._CPU utilization b._memory utilization c._disk I/O's (per second)
	C-MSS-68070	The MSS performance management application service shall be capable of receiving resource utilization information requests from the SDPS Client subsystem via the Management Agent Service.
	C-MSS-68050	The MSS performance management application service shall be capable of receiving resource utilization information requests from the SDPS Data Server subsystems via Management Agent Service.
TC020.010	C-ISS-04060	The portion of the DAAC LAN supporting the SDPS function of receiving science data shall contribute to the function's operational availability of 0.999 at a minimum and a mean down time of two (2) hours or less during times of staffed operation.
	C-ISS-04070	The portion of the DAAC LAN supporting the SDPS function of archiving and distributing data shall contribute to the function's operational availability of 0.98 at a minimum and a mean down time of two (2) hours or less during times of staffed operation.
	C-ISS-04080	The portion of the DAAC LAN supporting the SDPS function of User Interfaces to Client, Interoperability, Data Server, and Data Management (IMS) services at Individual DAAC Sites shall contribute to the function's operational availability of 0.993 at a minimum and a mean down time requirement of two (2) hours or less during times of staffed operations.
	C-ISS-04090	The portion of the DAAC LAN supporting the SDPS function of information searches on the ECS Directory shall contribute to the function's operational availability of 0.993 at a minimum and a mean down time of two (2) hours or less during times of staffed operation.
	C-ISS-04102	The portion of the EDC DAAC LAN supporting the SDPS function of Data Acquisition Request (DAR) Submittal including TOOs shall contribute to the function's operational availability of 0.993 at a minimum and mean down time of two (2) hours or less during times of staffed operation.

Table A-1. Requirements Verification Matrix (34 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-ISS-04110	The portion of the DAAC LAN supporting the SDPS function of metadata ingest and update shall contribute to the function's operational availability of 0.96 at a minimum and a mean down time of four (4) hours or less during times of staffed operation.
	C-ISS-04120	The portion of the DAAC LAN supporting the SDPS function of information searches on local holdings shall contribute to the function's operational availability of 0.96 at a minimum and a mean down time of four (4) hours or less during times of staffed operation.
	C-ISS-04130	The portion of the DAAC LAN supporting the SDPS function of local data order submission shall contribute to the function's operational availability of 0.96 at a minimum and a mean down time of four (4) hours or less during times of staffed operations.
	C-ISS-04140	The portion of the DAAC LAN supporting the SDPS function of local data order submission across DAACs shall contribute to the function's operational availability of 0.96 at a minimum and a mean down time of four (4) hours or less during times of staffed operation.
	C-ISS-04150	The portion of the DAAC LAN supporting the SDPS function of Client, Interoperability, Data Management and Data Server (IMS) Data Base Management and Maintenance Interface shall contribute to the function's operational availability of 0.96 at a minimum and a mean down time of four (4) hours or less during times of staffed operation.
	C-ISS-04160	The ISS elements and components shall include the on-line (operational mode) and off-line (test mode) fault detection and isolation capabilities required to achieve the specified operational availability requirements.
	C-ISS-04165	The maximum down time of the ISS-INHCI shall not exceed twice the required MDT in 99 percent of failure occurrences.
	C-ISS-04170	The EOC Operational LAN shall be configured to support the FOS availability of .99925 and a mean down time of < 5 minutes for non-critical real-time data during times of staffed operation.
	C-ISS-06000	The ISS network architecture shall enable expansion to GByte networks including the ability to provide increased volume of data distribution/access.
TC021.001	C-MSS-12005	The MSS Management User Interface (MUI) Service shall be compatible with the ECS management framework.
	C-MSS-12010	The MSS Management User Interface (MUI) Service shall provide a graphical user interface that is OSF/MOTIF compliant
	C-MSS-12020	The MSS MUI Service shall have the capability to respond to keyboard and mouse input devices
	C-MSS-12040	The MSS MUI Service shall provide a capability for an application to add/delete a symbol and to modify a symbol's shape, color and position

Table A-1. Requirements Verification Matrix (35 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-12090	The MSS MUI Service shall provide a capability for applications to establish a dialog session with the M&O Staff
	C-MSS-12140	The MSS MUI Service shall provide the capability for an application to register and unregister managed objects.
	C-MSS-12170	The MSS MUI Service shall provide the capability to register and unregister management applications.
	C-MSS-12180	The MSS MUI Service shall provide the capability for an application to display on-line help windows
	C-MSS-14010	The MSS Maps/Collection Service shall retain the status of managed objects and their relationship to symbols that comprise a graphical representation of the physical network topology.
	C-MSS-14020	The MSS Map/Collection Service shall provide a capability to define maps and objects.
	C-MSS-14030	The MSS Map/Collection Service shall provide a capability to define a hierarchical relationship between maps and sub-maps (i.e., a graphical hierarchical tree)
	C-MSS-16005	The ECS management protocol shall be the SNMP standard as specified in RFC 1157.
	C-MSS-16030	The MSS Monitor/Control Service shall be able to communicate via ECS management protocol with the MSS Management Agent Service to send ECS management set messages to configure and control the processing performed by the ECS management agent.
	C-MSS-20010	The MSS Discovery Service shall discover (via network protocol) new instances of managed objects.
	C-MSS-20040	The MSS Discovery Service shall update the object database after the Discovery Service receives a request to register/unregister a managed object.
	C-MSS-60010	The MSS Fault Management Application Service shall provide the capability to create and display graphical representations of a given network topology consisting of the following: a. routers b. communication lines c. hosts d. peripherals e. applications
	C-MSS-60020	The MSS Fault Management Application Service shall provide the capability to define categories of faults.
	C-MSS-60030	The MSS Fault Management Application Service shall provide the capability to assign faults to categories.
	C-MSS-60040	The MSS Fault Management Application Service shall provide the capability to assign severity levels to faults.
TC021.002	C-MSS-12005	The MSS Management User Interface (MUI) Service shall be compatible with the ECS management framework.

Table A-1. Requirements Verification Matrix (36 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-12010	The MSS Management User Interface (MUI) Service shall provide a graphical user interface that is OSF/MOTIF compliant
	C-MSS-12020	The MSS MUI Service shall have the capability to respond to keyboard and mouse input devices
	C-MSS-12080	The MSS MUI Service shall provide a capability for applications to alert the M&O Staff
	C-MSS-12090	The MSS MUI Service shall provide a capability for applications to establish a dialog session with the M&O Staff
	C-MSS-14040	The MSS Map/Collection Service shall propagate events associated with objects up the hierarchical tree
	C-MSS-16005	The ECS management protocol shall be the SNMP standard as specified in RFC 1157.
	C-MSS-16050	The MSS Monitor/Control Service shall allow customized M&O staff-event notifications and automatic actions.
	C-MSS-16070	The MSS Monitor/Control Service shall automatically report when a threshold has been exceeded by generating a ECS management event
	C-MSS-20020	The MSS Discovery Service shall detect missing occurrences of managed objects.
	C-MSS-20030	The MSS Discovery Service shall report missing occurrences of managed objects.
	C-MSS-60070	The MSS Fault Management Application Service shall provide the capability to specify additional information to be added to a disk log file, based on the fault category, when the notification of a fault is received.
	C-MSS-60100	The MSS Fault Management Application Service shall have the capability to poll for the detection of fault/performance information.
	C-MSS-60110	The MSS Fault Management Application Service shall be capable of receiving fault notifications.
	C-MSS-60150	The MSS Fault Management Application Service shall have the capability to receive fault notifications from the Management Agent Service.
TC021.003	C-MSS-60160	The MSS EMC Fault Management Application Service shall have the capability to receive notifications of detected faults and degradation of performance from: a. Site fault management applications b. Other external systems as defined in Section 5.1.
	C-MSS-60170	The MSS EMC Fault Management Application Service shall be capable of requesting fault notification and performance degradation data from : a. Site Fault Management Applications b. Other external systems as defined in Section 5.1.

Table A-1. Requirements Verification Matrix (37 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-60180	The MSS EMC Fault Management Application Service shall be capable of receiving summarized fault notification and performance degradation data from: a. Site fault management applications b. Other external systems as defined in Section 5.1.
	C-MSS-60190	The MSS Fault Management Application Service shall use the Logging Services to record each detected fault.
	C-MSS-60200	The MSS Fault Management Application Service shall have the capability to generate the following types of notifications for detected faults : a change in the color of an icon on a display a message in a pop-up notification window logging the following fault information to a disk log file: 1. fault type logging the following fault information to a disk log file 3. identification of the source of the notification 4. fault data received with the notification operator-defined descriptive text d. audible alert
	C-MSS-60220	The MSS Fault Management Application Service shall have the capability to send the notification of a fault to registered recipients.
	C-MSS-60230	The MSS Fault Management Application Service shall have the capability of generating a notification within a maximum of five minutes of fault detection.
	C-MSS-60340	The MSS Fault Management Application Service shall be capable of verifying the operational status of a host.
	ESN-0780#A	The network elements including the Internet interfaces, shall have the capability to report, periodically and on an interactive basis , network statistics to the ESN network management function, including the following information: a. Network round trip delay b. Network reset and restart indications c. Outages and CRC errors d. Performance statistics
	ESN-0810#A	ESN shall provide the following fault management functions at a minimum: a. detect the occurrence of faults, b. control the collection of fault information, and c. diagnose the probable cause of a detected fault

Table A-1. Requirements Verification Matrix (38 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD1703#A	ECS shall provide maintenance and operations interfaces to the DAACs to support the functions of: a). System Management b). Science Algorithm Integration c). Product Generation d). Data Archive/Distribution e). User Support Services f). System Maintenance
	SMC-3390#A	The SMC shall generate alert indicators of fault or degraded conditions with the corrective actions.
	SMC-3395#A	The LSM shall generate, in response to each limit check threshold, alert indicators of fault or degraded conditions.
TC021.004	C-MSS-60210	The MSS Fault Management Application Service shall maintain a list of external service providers, M&O operators, and applications to be notified in the event that a specified fault is detected.
	C-MSS-60400	The MSS Fault Management Application Service shall support, maintain, and update system fault management policies and procedures, to include: a. Fault Identification b. Fault priorities c. Recovery or corrective actions
	C-MSS-60410	The MSS Site Fault Management Application Service shall have the capability to receive Fault Management Policies and Procedures from the EMC.
	C-MSS-60500	The MSS EMC Fault Management Application Service shall coordinate the recovery from conditions of performance degradation and faults with the sites and external network service providers.
	C-MSS-60510	The MSS EMC Fault Management Application Service at the SMC shall coordinate, as necessary via directives and instructions, the recovery from faults reported from a site.
TC021.005	C-MSS-60420	The MSS Fault Management Application Service shall interface with the MSS Configuration Management Application Service and schedule a change in the configuration of the site when such a change in the configuration of the site is deemed necessary to recover from a fault.
	C-MSS-60520	The MSS Fault Management Application Service shall provide the capability to allow the specification and execution of action routines in response to the notification of a fault.
	C-MSS-60530	The MSS Fault Management Application Service shall provide the capability to pass parameters to action routines.
	C-MSS-60540	The MSS Fault Management Application Service shall utilize office automation support tools for the generation of directives and instructions for recovery from faults within its site.

Table A-1. Requirements Verification Matrix (39 of 134)

Test Case ID	Requirement ID	Requirement Text
TC021.006	C-MSS-12010	The MSS Management User Interface (MUI) Service shall provide a graphical user interface that is OSF/MOTIF compliant
	C-MSS-60600	The MSS Fault Management Application Service shall have the capability to generate, on an interactive and on a scheduled basis, reports on performance/error data that it has been configured to collect.
	C-MSS-60610	The MSS Fault Management Application Service shall have the capability to build histories for different types of errors and events detected, for the purpose of analysis.
	C-MSS-60620	The MSS Fault Management Application Service shall have the capability to redirect reports to: a. console b. disk file c. printer
	EOSD1703#A	ECS shall provide maintenance and operations interfaces to the DAACs to support the functions of: a). System Management b). Science Algorithm Integration c). Product Generation d). Data Archive/Distribution e). User Support Services f). System Maintenance
TC022.001	C-MSS-18340	The MSS Management Data Access Service shall provide the capability for an application to selectively read a record from a log file.
TC022.002	C-MSS-18340	The MSS Management Data Access Service shall provide the capability for an application to selectively read a record from a log file.
TC023.001	C-MSS-76000	The MSS accountability management service shall be capable of retrieving user activity data (user id, type of user activity, data items used (browsed, searched, or ordered), and date/time of activity) from records generated by the SDPS Data Server and Client subsystems.
TC023.002	C-MSS-76010	The MSS accountability management service shall be capable of querying via the Management Data Access service user activity data stored in the Management Database.
TC023.002	C-MSS-76020	The MSS accountability management service shall be capable of retrieving all activities associated with a particular user or data item via the Management Data Access service.
TC023.003	C-MSS-76040	The MSS Accountability Management Service shall be capable of reporting audit information to M&O staff via the MUI service.
TC023.004	C-MSS-77080	The MSS Accountability Management Service shall have the capability to generate reports from collected management data.

Table A-1. Requirements Verification Matrix (40 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-77090	The MSS Accountability Management Service shall have the capability to redirect reports: a._console b._disk file c._printer
TC024.002	C-MSS-40600	The MSS configuration management application service shall provide a capability with which to specify a need for ECS system changes, both for enhancing system capabilities and for correcting non-conformance with system requirements.
TC024.003	C-MSS-40600	The MSS configuration management application service shall provide a capability with which to specify a need for ECS system changes, both for enhancing system capabilities and for correcting non-conformance with system requirements.
TC024.004	C-MSS-40600	The MSS configuration management application service shall provide a capability with which to specify a need for ECS system changes, both for enhancing system capabilities and for correcting non-conformance with system requirements.
TC024.005	C-MSS-40690	The MSS configuration management application service at the SMC shall maintain the status of responses to change evaluation requests.
	C-MSS-40730	The MSS configuration management application service at the SMC shall maintain historical records of ECS configuration change requests, non-conformance reports, and system impact assessments.
	C-MSS-40750	The MSS configuration management application service at the SMC shall track approval and closure status of configuration change requests and non-conformance reports.
TC024.006	C-MSS-40690	The MSS configuration management application service at the SMC shall maintain the status of responses to change evaluation requests.
	C-MSS-40730	The MSS configuration management application service at the SMC shall maintain historical records of ECS configuration change requests, non-conformance reports, and system impact assessments.
	C-MSS-40750	The MSS configuration management application service at the SMC shall track approval and closure status of configuration change requests and non-conformance reports.
TC024.007	C-MSS-40610	The MSS configuration management application service shall store copies of non-conformance reports and requests to modify ECS components and configurations.
	C-MSS-40670	The MSS configuration management application service at the SMC shall receive and store impact assessments in response to change evaluation requests.

Table A-1. Requirements Verification Matrix (41 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-40690	The MSS configuration management application service at the SMC shall maintain the status of responses to change evaluation requests.
	C-MSS-40730	The MSS configuration management application service at the SMC shall maintain historical records of ECS configuration change requests, non-conformance reports, and system impact assessments.
TC024.008	C-MSS-40610	The MSS configuration management application service shall store copies of non-conformance reports and requests to modify ECS components and configurations.
	C-MSS-40670	The MSS configuration management application service at the SMC shall receive and store impact assessments in response to change evaluation requests.
	C-MSS-40690	The MSS configuration management application service at the SMC shall maintain the status of responses to change evaluation requests.
	C-MSS-40730	The MSS configuration management application service at the SMC shall maintain historical records of ECS configuration change requests, non-conformance reports, and system impact assessments.
TC024.009	C-MSS-40620	The MSS configuration management application service at the sites shall provide a capability with which to forward non-conformance reports and requests for ECS configuration changes to the SMC.
	C-MSS-40650	The MSS configuration management application service at the SMC shall receive configuration change requests and non-conformance reports in electronic form from the sites
	C-MSS-40660	The MSS configuration management application service at the SMC shall distribute change evaluation requests to designated organizations system-wide and record evaluation assignments and distribution status.
	C-MSS-40770	The MSS configuration management application service at the SMC shall collect, and make available system-wide, the allocations, schedules and status of tasks for implementing CCB-approved changes to ECS hardware and software and for correcting non-conformance with system requirements
TC024.010	C-MSS-40650	The MSS configuration management application service at the SMC shall receive configuration change requests and non-conformance reports in electronic form from the sites
	C-MSS-40660	The MSS configuration management application service at the SMC shall distribute change evaluation requests to designated organizations system-wide and record evaluation assignments and distribution status.

Table A-1. Requirements Verification Matrix (42 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-40720	The MSS configuration management application service at the SMC shall make non-conformance reports, configuration change requests, assessments, and status available for system-wide viewing.
	C-MSS-40760	The MSS configuration management application service at the SMC shall report, and make available system-wide lists of the identity and disposition of configuration change requests and non-conformance reports against ECS baselines.
	C-MSS-40770	The MSS configuration management application service at the SMC shall collect, and make available system-wide, the allocations, schedules and status of tasks for implementing CCB-approved changes to ECS hardware and software and for correcting non-conformance with system requirements
TC024.013	C-MSS-40670	The MSS configuration management application service at the SMC shall receive and store impact assessments in response to change evaluation requests.
	C-MSS-40680	The MSS configuration management service at the SMC shall electronically link impact assessments to their associated change requests.
	C-MSS-40700	The MSS configuration management application service at the SMC shall record summaries of impact assessments received.
BC007.001	DADS0901#A	The DADS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management h. Distribution and Ingest Management
	PGS-0310#A	The PGS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management c. Accounting Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management.
	EOSD0510#A	ECS shall be capable of being tested during all phases of its development and flight operations.

Table A-1. Requirements Verification Matrix (43 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD0740#A	Each ECS element shall provide a set of real or simulated functional capabilities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. ECS System (Integration of ECS elements)
	EOSD0750#A	Each ECS element shall provide a set of real or simulated functions which interfaces with both its ECS internal and external entities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. EOSDIS System (Integration of EOSDIS elements)
	EOSD0760#A	Each ECS element shall support end-to-end EOS system testing and fault isolation.
	EOSD0500#A	ECS shall perform the following major functions: a. EOS Mission Planning and Scheduling b. EOS Mission Operations c. Command and Control d. Communications and Networking e. Data Input f. Data Processing g. Data Storage h. Data Distribution i. Information Management j. End-to-End Fault Management k. System Management
	EOSD1703#A	ECS shall provide maintenance and operations interfaces to the DAACs to support the functions of: a). System Management b). Science Algorithm Integration c). Product Generation d). Data Archive/Distribution e). User Support Services f). System Maintenance
	NI-0460#A	ECS shall have the capability to receive periodic information regarding NOLAN network performance and link utilization.
	SMC-1345#A	The LSM shall perform priority management services to resolve conflicts for ECS resources.
	SMC-3345#A	The LSM shall perform quality assurance for its site/elements performance as well as programmatic areas that includes, at a minimum: a. Quality testing, benchmarks and audits for element enhancement implementations b. Quality checking and audits of products processed and delivered c. Quality testing and audits of element resource performance,

Table A-1. Requirements Verification Matrix (44 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-3355#A	The LSM shall implement the performance criteria from SMC (including parametric limits and operational threshold levels) for evaluating element resource performance.
	SMC-3375#A	For each limit checked parameter, the LSM (including those thresholds directed by the SMC) shall have the capability of evaluating multiple levels of thresholds including, at a minimum: a. On/off b. Pass/fail c. Various levels of degradation
	SMC-3385#A	The LSM shall evaluate system performance against the ESDIS project established performance criteria.
	SMC-3395#A	The LSM shall generate, in response to each limit check threshold, alert indicators of fault or degraded conditions.
	SMC-3397#A	The LSM shall generate, as needed, requests for performance testing, including, at a minimum: a. Resource to be tested b. Test purpose c. Requested test priority d. Required test environment e. Impacts to operations f. Expected test results
	SMC-3415#A	The LSM shall perform short and long-term trend analysis of element performance, including, at a minimum: a. Operational status b. Performance of a particular resource c. Maintenance activities (e.g., number of repairs per item)
	EOSD0545#A	ECS shall be able to accommodate growth (e.g., capacity) in all of its functions as well as the addition of new functions.
	EOSD3490#A	Reliability statistics for ECS shall be collected and monitored using the Mean Time Between Maintenance (MTBM) for each component and operational capability.
	EOSD3492#A	RMA data shall be maintained in a repository accessible for logistics analysis and other purposes.
	EOSD3620#A	ECS shall predict and periodically assess maintainability by measuring the actual MDT and comparing to the required MDT.
	EOSD3630#A	The maximum down time shall not exceed twice the required MDT in 99 percent of failure occurrences.

Table A-1. Requirements Verification Matrix (45 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD3700#A	ECS functions shall have an operational availability of 0.96 at a minimum (.998 design goal) and an MDT of four (4) hours or less (1.5 hour design goal), unless otherwise specified. The above requirement covers equipment including: a. "Non-critical" equipment configured with the critical equipment supporting the functional capabilities in the requirements. b. Equipment providing other functionality not explicitly stated in the RMA requirements that follow.
	EOSD3710#A	The ECS shall have no single point of failure for functions associated with real-time operations of the spacecraft and instruments.
	EOSD4035#A	The ESN shall have no single point of failure for functions associated with network databases and configuration data.
	EOSD4036#A	The operational availability of individual ESN segments shall be consistent with the specified operational availability of the supported ECS functions.
	EOSD5000#A	ECS shall enable the addition of other data providers, e.g. DAACs, SCFs, ADCs, ODCs, which may: - provide heterogeneous services, i.e. services in support of EOS which may be less than or different than ECS services. - be connected with varying topologies - have variable levels of reliability or operational availability.
	EOSD5070#A	ECS shall enable expansion to GByte networks including the ability to provide increased volume of data distribution/access.
	ESN-0815#A	Network simulation and traffic modeling capability shall be provided to troubleshoot network problems and to use in network planning.
	ESN-1060#A	The ESN performance management function shall provide the capability to evaluate the performance of ESN resources and interconnection activities.
	ESN-1065#A	The ESN performance management function shall include trend analysis for prediction of loading and bottlenecks/delays.
	ESN-1090#A	The ESN shall provide the capability to control the communications performance parameters of the network.
	ESN-1206#A	The ESN capacity and performance shall be consistent with the specified capacity and performance requirements of the ECS functions.
	ESN-1207#A	The ESN capacity and performance shall be capable of expansion to be consistent with the specified capacity and performance growth requirements of the ECS elements and functions.
	ESN-0240#A	The ESN shall be extensible in its design to provide capability for growth and enhancement.
	ESN-0740#A	The ESN network management service shall retrieve performance/fault data about ESN protocol stacks and equipment.

Table A-1. Requirements Verification Matrix (46 of 134)

Test Case ID	Requirement ID	Requirement Text
	ESN-0750#A	The ESN shall provide statistical processing capabilities to allow extraction and tabulation of network performance data.
	ESN-0780#A	The network elements including the Internet interfaces, shall have the capability to report, periodically and on an interactive basis, network statistics to the ESN network management function, including the following information: a. Network round trip delay b. Network reset and restart indications c. Outages and CRC errors d. Performance statistics
	EOSD3510#A	Reliability predictions shall be calculated in accordance with the parts count analysis method, Appendix A, of MIL-HDBK-217F, Reliability Prediction of Electronic Equipment.
	EOSD3600#A	Maintainability shall be predicted in accordance with MIL-HDBK-472, Maintainability Prediction, Procedure IV.
	EOSD3625#A	For ECS functions with a backup capability, ECS shall use switch over time to the backup capability in measuring maintainability, rather than down time, when the component goes down.
	EOSD0545#A	ECS shall be able to accommodate growth (e.g., capacity) in all of its functions as well as the addition of new functions.
	EOSD3490#A	Reliability statistics for ECS shall be collected and monitored using the Mean Time Between Maintenance (MTBM) for each component and operational capability.
	EOSD3492#A	RMA data shall be maintained in a repository accessible for logistics analysis and other purposes.
	EOSD3620#A	ECS shall predict and periodically assess maintainability by measuring the actual MDT and comparing to the required MDT.
	EOSD3630#A	The maximum down time shall not exceed twice the required MDT in 99 percent of failure occurrences.
	EOSD3700#A	ECS functions shall have an operational availability of 0.96 at a minimum (.998 design goal) and an MDT of four (4) hours or less (1.5 hour design goal), unless otherwise specified. The above requirement covers equipment including: a. "Non-critical" equipment configured with the critical equipment supporting the functional capabilities in the requirements. b. Equipment providing other functionality not explicitly stated in the RMA requirements that follow.
	EOSD3710#A	The ECS shall have no single point of failure for functions associated with real-time operations of the spacecraft and instruments.
	EOSD4035#A	The ESN shall have no single point of failure for functions associated with network databases and configuration data.

Table A-1. Requirements Verification Matrix (47 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD4036#A	The operational availability of individual ESN segments shall be consistent with the specified operational availability of the supported ECS functions.
	EOSD5000#A	ECS shall enable the addition of other data providers, e.g. DAACs, SCFs, ADCs, ODCs, which may: - provide heterogeneous services, i.e. services in support of EOS which may be less than or different than ECS services. - be connected with varying topologies - have variable levels of reliability or operational availability.
	EOSD5070#A	ECS shall enable expansion to GByte networks including the ability to provide increased volume of data distribution/access.
	ESN-0815#A	Network simulation and traffic modeling capability shall be provided to troubleshoot network problems and to use in network planning.
	ESN-1060#A	The ESN performance management function shall provide the capability to evaluate the performance of ESN resources and interconnection activities.
	ESN-1065#A	The ESN performance management function shall include trend analysis for prediction of loading and bottlenecks/delays.
	ESN-1090#A	The ESN shall provide the capability to control the communications performance parameters of the network.
	ESN-1206#A	The ESN capacity and performance shall be consistent with the specified capacity and performance requirements of the ECS functions.
	ESN-1207#A	The ESN capacity and performance shall be capable of expansion to be consistent with the specified capacity and performance growth requirements of the ECS elements and functions.
	ESN-0240#A	The ESN shall be extensible in its design to provide capability for growth and enhancement.
	ESN-0740#A	The ESN network management service shall retrieve performance/fault data about ESN protocol stacks and equipment.
	ESN-0750#A	The ESN shall provide statistical processing capabilities to allow extraction and tabulation of network performance data.
	ESN-0780#A	The network elements including the Internet interfaces, shall have the capability to report, periodically and on an interactive basis, network statistics to the ESN network management function, including the following information: a. Network round trip delay b. Network reset and restart indications c. Outages and CRC errors d. Performance statistics
	EOSD3510#A	Reliability predictions shall be calculated in accordance with the parts count analysis method, Appendix A, of MIL-HDBK-217F, Reliability Prediction of Electronic Equipment.

Table A-1. Requirements Verification Matrix (48 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD3600#A	Maintainability shall be predicted in accordance with MIL-HDBK-472, Maintainability Prediction, Procedure IV.
	EOSD3625#A	For ECS functions with a backup capability, ECS shall use switchover time to the backup capability in measuring maintainability, rather than down time, when the component goes down.
BC007.002	DADS0901#A	The DADS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management h. Distribution and Ingest Management
	PGS-0310#A	The PGS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management c. Accounting Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management.
	EOSD0510#A	ECS shall be capable of being tested during all phases of its development and flight operations.
	EOSD0740#A	Each ECS element shall provide a set of real or simulated functional capabilities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. ECS System (Integration of ECS elements)
	EOSD0750#A	Each ECS element shall provide a set of real or simulated functions which interfaces with both its ECS internal and external entities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. EOSDIS System (Integration of EOSDIS elements)
	EOSD0760#A	Each ECS element shall support end-to-end EOS system testing and fault isolation.

Table A-1. Requirements Verification Matrix (49 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD0500#A	ECS shall perform the following major functions: a. EOS Mission Planning and Scheduling b. EOS Mission Operations c. Command and Control d. Communications and Networking e. Data Input f. Data Processing g. Data Storage h. Data Distribution i. Information Management j. End-to-End Fault Management k. System Management
	EOSD1703#A	ECS shall provide maintenance and operations interfaces to the DAACs to support the functions of: a). System Management b). Science Algorithm Integration c). Product Generation d). Data Archive/Distribution e). User Support Services f). System Maintenance
	NI-0460#A	ECS shall have the capability to receive periodic information regarding NOLAN network performance and link utilization.
	SMC-1345#A	The LSM shall perform priority management services to resolve conflicts for ECS resources.
	SMC-3345#A	The LSM shall perform quality assurance for its site/elements performance as well as programmatic areas that includes, at a minimum: a. Quality testing, benchmarks and audits for element enhancement implementations b. Quality checking and audits of products processed and delivered c. Quality testing and audits of element resource performance,
	SMC-3355#A	The LSM shall implement the performance criteria from SMC (including parametric limits and operational threshold levels) for evaluating element resource performance.
	SMC-3375#A	For each limit checked parameter, the LSM (including those thresholds directed by the SMC) shall have the capability of evaluating multiple levels of thresholds including, at a minimum: a. On/off b. Pass/fail c. Various levels of degradation
	SMC-3385#A	The LSM shall evaluate system performance against the ESDIS project established performance criteria.
	SMC-3395#A	The LSM shall generate, in response to each limit check threshold, alert indicators of fault or degraded conditions.

Table A-1. Requirements Verification Matrix (50 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-3397#A	The LSM shall generate, as needed, requests for performance testing, including, at a minimum: a. Resource to be tested b. Test purpose c. Requested test priority d. Required test environment e. Impacts to operations f. Expected test results
	SMC-3415#A	The LSM shall perform short and long-term trend analysis of element performance, including, at a minimum: a. Operational status b. Performance of a particular resource c. Maintenance activities (e.g., number of repairs per item)
	EOSD0545#A	ECS shall be able to accommodate growth (e.g., capacity) in all of its functions as well as the addition of new functions.
	EOSD3490#A	Reliability statistics for ECS shall be collected and monitored using the Mean Time Between Maintenance (MTBM) for each component and operational capability.
	EOSD3492#A	RMA data shall be maintained in a repository accessible for logistics analysis and other purposes.
	EOSD3620#A	ECS shall predict and periodically assess maintainability by measuring the actual MDT and comparing to the required MDT.
	EOSD3630#A	The maximum down time shall not exceed twice the required MDT in 99 percent of failure occurrences.
	EOSD3700#A	ECS functions shall have an operational availability of 0.96 at a minimum (.998 design goal) and an MDT of four (4) hours or less (1.5 hour design goal), unless otherwise specified. The above requirement covers equipment including: a. "Non-critical" equipment configured with the critical equipment supporting the functional capabilities in the requirements. b. Equipment providing other functionality not explicitly stated in the RMA requirements that follow.
	EOSD3710#A	The ECS shall have no single point of failure for functions associated with real-time operations of the spacecraft and instruments.
	EOSD4035#A	The ESN shall have no single point of failure for functions associated with network databases and configuration data.
	EOSD4036#A	The operational availability of individual ESN segments shall be consistent with the specified operational availability of the supported ECS functions.

Table A-1. Requirements Verification Matrix (51 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD5000#A	ECS shall enable the addition of other data providers, e.g. DAACs, SCFs, ADCs, ODCs, which may: - provide heterogeneous services, i.e. services in support of EOS which may be less than or different than ECS services. - be connected with varying topologies - have variable levels of reliability or operational availability.
	EOSD5070#A	ECS shall enable expansion to GByte networks including the ability to provide increased volume of data distribution/access.
	ESN-0815#A	Network simulation and traffic modeling capability shall be provided to troubleshoot network problems and to use in network planning.
	ESN-1060#A	The ESN performance management function shall provide the capability to evaluate the performance of ESN resources and interconnection activities.
	ESN-1065#A	The ESN performance management function shall include trend analysis for prediction of loading and bottlenecks/delays.
	ESN-1090#A	The ESN shall provide the capability to control the communications performance parameters of the network.
	ESN-1206#A	The ESN capacity and performance shall be consistent with the specified capacity and performance requirements of the ECS functions.
	ESN-1207#A	The ESN capacity and performance shall be capable of expansion to be consistent with the specified capacity and performance growth requirements of the ECS elements and functions.
	ESN-0240#A	The ESN shall be extensible in its design to provide capability for growth and enhancement.
	ESN-0740#A	The ESN network management service shall retrieve performance/fault data about ESN protocol stacks and equipment.
	ESN-0750#A	The ESN shall provide statistical processing capabilities to allow extraction and tabulation of network performance data.
	ESN-0780#A	The network elements including the Internet interfaces, shall have the capability to report, periodically and on an interactive basis, network statistics to the ESN network management function, including the following information: a. Network round trip delay b. Network reset and restart indications c. Outages and CRC errors d. Performance statistics
	EOSD3510#A	Reliability predictions shall be calculated in accordance with the parts count analysis method, Appendix A, of MIL-HDBK-217F, Reliability Prediction of Electronic Equipment.
	EOSD3600#A	Maintainability shall be predicted in accordance with MIL-HDBK-472, Maintainability Prediction, Procedure IV.

Table A-1. Requirements Verification Matrix (52 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD3625#A	For ECS functions with a backup capability, ECS shall use switchover time to the backup capability in measuring maintainability, rather than down time, when the component goes down.
BC007.004	DADS0901#A	The DADS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management h. Distribution and Ingest Management
	PGS-0310#A	The PGS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management c. Accounting Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management.
	EOSD0510#A	ECS shall be capable of being tested during all phases of its development and flight operations.
	EOSD0740#A	Each ECS element shall provide a set of real or simulated functional capabilities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. ECS System (Integration of ECS elements)
	EOSD0750#A	Each ECS element shall provide a set of real or simulated functions which interfaces with both its ECS internal and external entities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. EOSDIS System (Integration of EOSDIS elements)
	EOSD0760#A	Each ECS element shall support end-to-end EOS system testing and fault isolation.

Table A-1. Requirements Verification Matrix (53 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD0500#A	ECS shall perform the following major functions: a. EOS Mission Planning and Scheduling b. EOS Mission Operations c. Command and Control d. Communications and Networking e. Data Input f. Data Processing g. Data Storage h. Data Distribution i. Information Management j. End-to-End Fault Management k. System Management
	EOSD1703#A	ECS shall provide maintenance and operations interfaces to the DAACs to support the functions of: a). System Management b). Science Algorithm Integration c). Product Generation d). Data Archive/Distribution e). User Support Services f). System Maintenance
	NI-0460#A	ECS shall have the capability to receive periodic information regarding NOLAN network performance and link utilization.
	SMC-1345#A	The LSM shall perform priority management services to resolve conflicts for ECS resources.
	SMC-3345#A	The LSM shall perform quality assurance for its site/elements performance as well as programmatic areas that includes, at a minimum: a. Quality testing, benchmarks and audits for element enhancement implementations b. Quality checking and audits of products processed and delivered c. Quality testing and audits of element resource performance,
	SMC-3355#A	The LSM shall implement the performance criteria from SMC (including parametric limits and operational threshold levels) for evaluating element resource performance.
	SMC-3375#A	For each limit checked parameter, the LSM (including those thresholds directed by the SMC) shall have the capability of evaluating multiple levels of thresholds including, at a minimum: a. On/off b. Pass/fail c. Various levels of degradation
	SMC-3385#A	The LSM shall evaluate system performance against the ESDIS project established performance criteria.
	SMC-3395#A	The LSM shall generate, in response to each limit check threshold, alert indicators of fault or degraded conditions.

Table A-1. Requirements Verification Matrix (54 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-3397#A	The LSM shall generate, as needed, requests for performance testing, including, at a minimum: a. Resource to be tested b. Test purpose c. Requested test priority d. Required test environment e. Impacts to operations f. Expected test results
	SMC-3415#A	The LSM shall perform short and long-term trend analysis of element performance, including, at a minimum: a. Operational status b. Performance of a particular resource c. Maintenance activities (e.g., number of repairs per item)
	EOSD0545#A	ECS shall be able to accommodate growth (e.g., capacity) in all of its functions as well as the addition of new functions.
	EOSD3490#A	Reliability statistics for ECS shall be collected and monitored using the Mean Time Between Maintenance (MTBM) for each component and operational capability.
	EOSD3492#A	RMA data shall be maintained in a repository accessible for logistics analysis and other purposes.
	EOSD3620#A	ECS shall predict and periodically assess maintainability by measuring the actual MDT and comparing to the required MDT.
	EOSD3630#A	The maximum down time shall not exceed twice the required MDT in 99 percent of failure occurrences.
	EOSD3700#A	ECS functions shall have an operational availability of 0.96 at a minimum (.998 design goal) and an MDT of four (4) hours or less (1.5 hour design goal), unless otherwise specified. The above requirement covers equipment including: a. "Non-critical" equipment configured with the critical equipment supporting the functional capabilities in the requirements. b. Equipment providing other functionality not explicitly stated in the RMA requirements that follow.
	EOSD3710#A	The ECS shall have no single point of failure for functions associated with real-time operations of the spacecraft and instruments.
	EOSD4035#A	The ESN shall have no single point of failure for functions associated with network databases and configuration data.
	EOSD4036#A	The operational availability of individual ESN segments shall be consistent with the specified operational availability of the supported ECS functions.

Table A-1. Requirements Verification Matrix (55 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD5000#A	ECS shall enable the addition of other data providers, e.g. DAACs, SCFs, ADCs, ODCs, which may: - provide heterogeneous services, i.e. services in support of EOS which may be less than or different than ECS services. - be connected with varying topologies - have variable levels of reliability or operational availability.
	EOSD5070#A	ECS shall enable expansion to GByte networks including the ability to provide increased volume of data distribution/access.
	ESN-0815#A	Network simulation and traffic modeling capability shall be provided to troubleshoot network problems and to use in network planning.
	ESN-1060#A	The ESN performance management function shall provide the capability to evaluate the performance of ESN resources and interconnection activities.
	ESN-1065#A	The ESN performance management function shall include trend analysis for prediction of loading and bottlenecks/delays.
	ESN-1090#A	The ESN shall provide the capability to control the communications performance parameters of the network.
	ESN-1206#A	The ESN capacity and performance shall be consistent with the specified capacity and performance requirements of the ECS functions.
	ESN-1207#A	The ESN capacity and performance shall be capable of expansion to be consistent with the specified capacity and performance growth requirements of the ECS elements and functions.
	ESN-0240#A	The ESN shall be extensible in its design to provide capability for growth and enhancement.
	ESN-0740#A	The ESN network management service shall retrieve performance/fault data about ESN protocol stacks and equipment.
	ESN-0750#A	The ESN shall provide statistical processing capabilities to allow extraction and tabulation of network performance data.
	ESN-0780#A	The network elements including the Internet interfaces, shall have the capability to report, periodically and on an interactive basis, network statistics to the ESN network management function, including the following information: a. Network round trip delay b. Network reset and restart indications c. Outages and CRC errors d. Performance statistics
	EOSD3510#A	Reliability predictions shall be calculated in accordance with the parts count analysis method, Appendix A, of MIL-HDBK-217F, Reliability Prediction of Electronic Equipment.
	EOSD3600#A	Maintainability shall be predicted in accordance with MIL-HDBK-472, Maintainability Prediction, Procedure IV.

Table A-1. Requirements Verification Matrix (56 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-6325#A	The LSM shall perform, as needed, data and user audit trails within its element.
	SMC-6335#A	The LSM shall, as needed, maintain and update a data tracking system that, at a minimum: a. Tracks data transport from element input to element output b. Allows the status of all product-production activities to be determined
	ESN-0775#A	The ESN management service shall have the capability to redirect its reports to different devices such as console, disk or printer.
	EOSD3625#A	For ECS functions with a backup capability, ECS shall use switchover time to the backup capability in measuring maintainability, rather than down time, when the component goes down.
BC007.005	DADS0901#A	The DADS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management h. Distribution and Ingest Management
	PGS-0310#A	The PGS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management c. Accounting Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management.
	EOSD0510#A	ECS shall be capable of being tested during all phases of its development and flight operations.
	EOSD0740#A	Each ECS element shall provide a set of real or simulated functional capabilities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. ECS System (Integration of ECS elements)
	EOSD0750#A	Each ECS element shall provide a set of real or simulated functions which interfaces with both its ECS internal and external entities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. EOSDIS System (Integration of EOSDIS elements)

Table A-1. Requirements Verification Matrix (57 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD0760#A	Each ECS element shall support end-to-end EOS system testing and fault isolation.
	EOSD0500#A	ECS shall perform the following major functions: a. EOS Mission Planning and Scheduling b. EOS Mission Operations c. Command and Control d. Communications and Networking e. Data Input f. Data Processing g. Data Storage h. Data Distribution i. Information Management j. End-to-End Fault Management k. System Management
	EOSD1703#A	ECS shall provide maintenance and operations interfaces to the DAACs to support the functions of: a). System Management b). Science Algorithm Integration c). Product Generation d). Data Archive/Distribution e). User Support Services f). System Maintenance
	NI-0460#A	ECS shall have the capability to receive periodic information regarding NOLAN network performance and link utilization.
	SMC-1345#A	The LSM shall perform priority management services to resolve conflicts for ECS resources.
	SMC-3345#A	The LSM shall perform quality assurance for its site/elements performance as well as programmatic areas that includes, at a minimum: a. Quality testing, benchmarks and audits for element enhancement implementations b. Quality checking and audits of products processed and delivered c. Quality testing and audits of element resource performance,
	SMC-3355#A	The LSM shall implement the performance criteria from SMC (including parametric limits and operational threshold levels) for evaluating element resource performance.
	SMC-3375#A	For each limit checked parameter, the LSM (including those thresholds directed by the SMC) shall have the capability of evaluating multiple levels of thresholds including, at a minimum: a. On/off b. Pass/fail c. Various levels of degradation
	SMC-3385#A	The LSM shall evaluate system performance against the ESDIS project established performance criteria.

Table A-1. Requirements Verification Matrix (58 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-3395#A	The LSM shall generate, in response to each limit check threshold, alert indicators of fault or degraded conditions.
	SMC-3397#A	The LSM shall generate, as needed, requests for performance testing, including, at a minimum: a. Resource to be tested b. Test purpose c. Requested test priority d. Required test environment e. Impacts to operations f. Expected test results
	SMC-3415#A	The LSM shall perform short and long-term trend analysis of element performance, including, at a minimum: a. Operational status b. Performance of a particular resource c. Maintenance activities (e.g., number of repairs per item)
	EOSD0545#A	ECS shall be able to accommodate growth (e.g., capacity) in all of its functions as well as the addition of new functions.
	EOSD3490#A	Reliability statistics for ECS shall be collected and monitored using the Mean Time Between Maintenance (MTBM) for each component and operational capability.
	EOSD3492#A	RMA data shall be maintained in a repository accessible for logistics analysis and other purposes.
	EOSD3620#A	ECS shall predict and periodically assess maintainability by measuring the actual MDT and comparing to the required MDT.
	EOSD3630#A	The maximum down time shall not exceed twice the required MDT in 99 percent of failure occurrences.
	EOSD3700#A	ECS functions shall have an operational availability of 0.96 at a minimum (.998 design goal) and an MDT of four (4) hours or less (1.5 hour design goal), unless otherwise specified. The above requirement covers equipment including: a. "Non-critical" equipment configured with the critical equipment supporting the functional capabilities in the requirements. b. Equipment providing other functionality not explicitly stated in the RMA requirements that follow.
	EOSD3710#A	The ECS shall have no single point of failure for functions associated with real-time operations of the spacecraft and instruments.
	EOSD4035#A	The ESN shall have no single point of failure for functions associated with network databases and configuration data.
	EOSD4036#A	The operational availability of individual ESN segments shall be consistent with the specified operational availability of the supported ECS functions.

Table A-1. Requirements Verification Matrix (59 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD5000#A	ECS shall enable the addition of other data providers, e.g. DAACs, SCFs, ADCs, ODCs, which may: - provide heterogeneous services, i.e. services in support of EOS which may be less than or different than ECS services. - be connected with varying topologies - have variable levels of reliability or operational availability.
	EOSD5070#A	ECS shall enable expansion to GByte networks including the ability to provide increased volume of data distribution/access.
	ESN-0815#A	Network simulation and traffic modeling capability shall be provided to troubleshoot network problems and to use in network planning.
	ESN-1060#A	The ESN performance management function shall provide the capability to evaluate the performance of ESN resources and interconnection activities.
	ESN-1065#A	The ESN performance management function shall include trend analysis for prediction of loading and bottlenecks/delays.
	ESN-1090#A	The ESN shall provide the capability to control the communications performance parameters of the network.
	ESN-1206#A	The ESN capacity and performance shall be consistent with the specified capacity and performance requirements of the ECS functions.
	ESN-1207#A	The ESN capacity and performance shall be capable of expansion to be consistent with the specified capacity and performance growth requirements of the ECS elements and functions.
	ESN-0240#A	The ESN shall be extensible in its design to provide capability for growth and enhancement.
	ESN-0740#A	The ESN network management service shall retrieve performance/fault data about ESN protocol stacks and equipment.
	ESN-0750#A	The ESN shall provide statistical processing capabilities to allow extraction and tabulation of network performance data.
	ESN-0780#A	The network elements including the Internet interfaces, shall have the capability to report, periodically and on an interactive basis, network statistics to the ESN network management function, including the following information: a. Network round trip delay b. Network reset and restart indications c. Outages and CRC errors d. Performance statistics
	EOSD3510#A	Reliability predictions shall be calculated in accordance with the parts count analysis method, Appendix A, of MIL-HDBK-217F, Reliability Prediction of Electronic Equipment.
	EOSD3600#A	Maintainability shall be predicted in accordance with MIL-HDBK-472, Maintainability Prediction, Procedure IV.

Table A-1. Requirements Verification Matrix (60 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD3625#A	For ECS functions with a backup capability, ECS shall use switchover time to the backup capability in measuring maintainability, rather than down time, when the component goes down.
TC025.001	C-CSS-22000	The CSS Message service shall provide an API for senders to send messages to receivers asynchronously without waiting for the receivers to receive it.
	C-CSS-22130	The CSS Message service shall associate the receiver to a returned value and maintain that information locally until the sender requests that information.
	C-CSS-22140	The CSS Message Service shall provide an API for the sender of the message to receive return information stored at the message queue.
TC025.002	C-CSS-22010	The CSS Message service shall provide an API for senders to send messages to receivers in a deferred synchronously manner through an intermediary where by they can contact the intermediary at a latter time to receive the result.
	C-CSS-22130	The CSS Message service shall associate the receiver to a returned value and maintain that information locally until the sender requests that information.
	C-CSS-22140	The CSS Message Service shall provide an API for the sender of the message to receive return information stored at the message queue.
	C-CSS-22090	The CSS Message Service shall provide the capability to locate and send (push model) the messages to receivers.
	C-CSS-22100	The CSS Message Service shall provide a non blocking API for the receiver to contact the message queue and get (pull model) the message.
	C-CSS-22120	The CSS Message service shall provide an API for the sender of the message to get the acknowledgment information the message service receives from the receivers.
	C-CSS-22110	The CSS Message service shall support guaranteed delivery of the message to the receiver.
TC025.003	C-CSS-22000	The CSS Message service shall provide an API for senders to send messages to receivers asynchronously without waiting for the receivers to receive it.
	C-CSS-22130	The CSS Message service shall associate the receiver to a returned value and maintain that information locally until the sender requests that information.
	C-CSS-22140	The CSS Message Service shall provide an API for the sender of the message to receive return information stored at the message queue.

Table A-1. Requirements Verification Matrix (61 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-CSS-22150	The CSS Message Service shall defer sending a message to a receiver, if the receiver is not active, and should try sending the message periodically with a set interval of time until the receiver is active.
TC025.004	C-CSS-22000	The CSS Message service shall provide an API for senders to send messages to receivers asynchronously without waiting for the receivers to receive it.
	C-CSS-22130	The CSS Message service shall associate the receiver to a returned value and maintain that information locally until the sender requests that information.
	C-CSS-22140	The CSS Message Service shall provide an API for the sender of the message to receive return information stored at the message queue.
	C-CSS-22150	The CSS Message Service shall defer sending a message to a receiver, if the receiver is not active, and should try sending the message periodically with a set interval of time until the receiver is active.
	C-CSS-22065	The CSS Message Service shall log event messages to the MSS management agents whenever the message service could not deliver a message to any receiver in the time period set by the sender of the message.
TC025.005	C-CSS-22000	The CSS Message service shall provide an API for senders to send messages to receivers asynchronously without waiting for the receivers to receive it.
	C-CSS-22130	The CSS Message service shall associate the receiver to a returned value and maintain that information locally until the sender requests that information.
	C-CSS-22140	The CSS Message Service shall provide an API for the sender of the message to receive return information stored at the message queue.
	C-CSS-22120	The CSS Message service shall provide an API for the sender of the message to get the acknowledgment information the message service receives from the receivers.
TC025.006	C-CSS-22010	The CSS Message service shall provide an API for senders to send messages to receivers in a deferred synchronously manner through an intermediary where by they can contact the intermediary at a latter time to receive the result.
	C-CSS-22130	The CSS Message service shall associate the receiver to a returned value and maintain that information locally until the sender requests that information.
BC008.002	ESN—0450#A	The ESN shall provide process-to-process communication service.

Table A-1. Requirements Verification Matrix (62 of 134)

Test Case ID	Requirement ID	Requirement Text
TC027.001	C-MSS-60030	The MSS Fault Management Application Service shall provide the capability to assign faults to categories._
	C-MSS-60020	The MSS Fault Management Application Service shall provide the capability to define categories of faults.
	C-MSS-60190	The MSS Fault Management Application Service shall use the Logging Services to record each detected fault.
	C-MSS-60040	The MSS Fault Management Application Service shall provide the capability to assign severity levels to faults._
	C-MSS-60060	The MSS Fault Management Application Service shall provide the capability to enable or disable the display of fault notifications received from a specific managed object based on fault category assigned to that fault._
	C-MSS-60070	The MSS Fault Management Application Service shall provide the capability to specify additional information to be added to a disk log file, based on the fault category, when the notification of a fault is received._
	C-MSS-60050	The MSS Fault Management Application Service shall be capable of providing the Management Data Access Service with a configurable list of fault categories that specify whether to enable or disable the logging of fault notifications for that fault category._
TC027.002	C-MSS-60610	The MSS Fault Management Application Service shall have the capability to build histories for different types of errors and events detected, for the purpose of analysis.
	C-MSS-60140	The MSS Site Fault Management Application Service shall have the capability to generate a fault notification when a predefined threshold on a performance metric is exceeded.
	C-MSS-60620	The MSS Fault Management Application Service shall have the capability to redirect reports to: a._console b._disk file c._printer
	C-MSS-60600	The MSS Fault Management Application Service shall have the capability to generate, on an interactive and on a scheduled basis, reports on performance/error data that it has been configured to collect.
	C-MSS-60120	The MSS Fault Management Application Service shall have the capability to define the frequency with which polling is done for the detection of fault/performance information._

Table A-1. Requirements Verification Matrix (63 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-60130	The MSS Fault Management Application Service shall provide the capability to detect the following types of faults, errors and events: _a._communications software version mismatch errors _b._communication software configuration errors _c._the following errors in communication hardware: __1. host not reachable __2. router not reachable __3. errors and failures of communication links _d._Errors in the communications protocols supported _e._degradation of performance due to established thresholds being exceeded _f._Peripherals _g._Databases _h._Applications: __1. process missing (Application or COTS product) __2. process in a loop __3. process failed
	C-MSS-60080	The MSS Fault Management Application Service shall have the capability to establish, view, modify and delete thresholds on performance metrics it measures._
	C-MSS-60100	The MSS Fault Management Application Service shall have the capability to poll for the detection of fault/performance information._
TC027.003	C-MSS-60160	The MSS EMC Fault Management Application Service shall have the capability to receive notifications of detected faults and degradation of performance from: a._Site fault management applications b._Other external systems as defined in Section 5.1.
	C-MSS-60180	The MSS EMC Fault Management Application Service shall be capable of receiving summarized fault notification and performance degradation data from: a._Site fault management applications b._Other external systems as defined in Section 5.1.
	C-MSS-60150	The MSS Fault Management Application Service shall have the capability to receive fault notifications from the Management Agent Service.
	C-MSS-60170	The MSS EMC Fault Management Application Service shall be capable of requesting fault notification and performance degradation data from : _a._Site Fault Management Applications _b._Other external systems as defined in Section 5.1.
	C-MSS-60110	The MSS Fault Management Application Service shall be capable of receiving fault notifications.

Table A-1. Requirements Verification Matrix (64 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-60400	The MSS EMC Fault Management Application Service shall support, maintain, and update system fault management policies and procedures, to include: _a._Fault Identification _b._Fault priorities _c._Recovery or corrective actions
TC027.004	C-MSS-60210	The MSS Fault Management Application Service shall maintain a list of external service providers, M&O operators, and applications to be notified in the event that a specified fault is detected.
	C-MSS-60220	The MSS Fault Management Application Service shall have the capability to send the notification of a fault to registered recipients.
	C-MSS-60230	The MSS Fault Management Application Service shall have the capability of generating a notification within a maximum of five minutes of fault detection.
	C-MSS-60200	The MSS Fault Management Application Service shall have the capability to generate the following types of notifications for detected faults : _a._a change in the color of an icon on a display _b._a message in a pop-up notification window _c._logging the following fault information to a disk log file: __1. fault type __2. date and time of occurrence of the fault __3. identification of the source of the notification (e.g. IP address, process name, etc.) __4. fault data received with the notification __5. operator-defined descriptive text _d._audible alert
	C-MSS-60395	The MSS Fault Management Application Service shall be capable of retrieving records of detected fault.
TC028.001	C-MSS-70010	The MSS site Security Management Application Service shall provide the capability to create, modify and delete user accounts with the following attributes: username password group identification code user identification code login directory command line interpreter
	C-MSS-70020	The MSS site Security Management Application Service shall enable the assignment of user accounts to groups based on the group identification code.
	C-MSS-70100	The MSS site Security Management Application Service shall provide the capability to set, maintain, and update access control information for ECS resources.

Table A-1. Requirements Verification Matrix (65 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-70110	The MSS site Security Management Application Service shall provide the capability to specify privileges for authorized users or user groups for access to ECS resources.
	C-MSS-70120	The MSS site Security Management Application Service shall provide the mechanism, for each ECS host, to allow or deny incoming requests from specific hosts to services.
	C-MSS-70450	The MSS site Security Management Application Service shall have the capability to detect the following types of intrusions: Login failures Unauthorized access to ECS resources Break-ins
TC028.002	C-MSS-70300	The MSS site Security Management Application Service shall have the capability to perform the following types of security tests: password auditing file system integrity checking auditing of user privileges auditing of resource access control information
	C-MSS-70430	The MSS site Security Management Application Service shall provide the capability to designate a user or group of users to receive a notification upon the detection of an intrusion.
	C-MSS-70400	The MSS site Security Management Application Service shall provide the capability to receive notifications of security events from the site Security Management Application Services.
TC028.003	C-MSS-70310	The MSS site Security Management Application Service shall provide the capability to perform security testing on a periodic and on an interactive basis.
	C-MSS-70320	The MSS site Security Management Application Service shall have the capability to send the results of the tests to the EMC Security Management Application Service.
	C-MSS-70430	The MSS site Security Management Application Service shall provide the capability to designate a user or group of users to receive a notification upon the detection of an intrusion.
	C-MSS-70440	The MSS site Security Management Application Service shall provide the capability to notify designated M&O staff(s) upon the detection of an intrusion.
	C-MSS-70460	The MSS site Security Management Application Service shall provide the capability of generating a notification within a maximum of five minutes of the detection of an intrusion.
TC028.004	C-MSS-70510	The MSS site Security Management Application Service shall upon detection of a compromise, isolate the compromised input I/O and the compromised area's output I/O until the compromise has been eliminated.

Table A-1. Requirements Verification Matrix (66 of 134)

Test Case ID	Requirement ID	Requirement Text
TC028.005	C-MSS-70130	The MSS site Security Management Application Service shall provide a command line interface and a GUI for the management of the following security databases: Authentication database Authorization database Network database
	C-MSS-70700	The MSS site Security Management Application Service shall have the capability to generate intrusion reports on the following: Login failures Unauthorized access to ECS resources Break-ins
	C-MSS-70710	The MSS site Security Management Application Service shall provide the capability to generate reports from collected management data.
	C-MSS-70720	The MSS site Security Management Application Service shall have the capability to redirect reports to: console disk file printer
TC030.001	C-MSS-18260	The MSS Management Data Access Service shall have the capability to schedule the transfer and loading log files into the management database at the site.
	C-MSS-18270	The MSS Management Data Access Service shall have the capability to schedule the archiving of log files at the site.
TC030.002	C-MSS-18040	The MSS Management Data Access Service shall maintain the integrity of the management database.
	C-MSS-18060	The MSS Management Data Access Service shall provide the capability for an application to access management data.
	C-MSS-18070	The MSS Management Data Access Service shall provide the capability to selectively access management data.
	C-MSS-18200	The MSS Management Data Access Service shall provide the capability for an application via APIs to update fields in the management database.
	C-MSS-18220	The MSS Management Data Access Service shall provide the capability for an application via APIs to alter tables and fields in the management database.
	C-MSS-18260	The MSS Management Data Access Service shall have the capability to schedule the transfer and loading log files into the management database at the site.
	C-MSS-18350	The MSS Management Data Access Service shall provide the capability for an application to load log files into the management database at the site

Table A-1. Requirements Verification Matrix (67 of 134)

Test Case ID	Requirement ID	Requirement Text
TC030.003	C-MSS-18340	The MSS Management Data Access Service shall provide the capability for an application to selectively read a record from a log file.
	C-MSS-18350	The MSS Management Data Access Service shall provide the capability for an application to load log files into the management database at the site
	C-MSS-18260	The MSS Management Data Access Service shall have the capability to schedule the transfer and loading log files into the management database at the site.
	C-MSS-18220	The MSS Management Data Access Service shall provide the capability for an application via APIs to alter tables and fields in the management database.
	C-MSS-18270	The MSS Management Data Access Service shall have the capability to schedule the archiving of log files at the site.
	C-MSS-18350	The MSS Management Data Access Service shall provide the capability for an application to load log files into the management database at the site
	C-MSS-18280	The MSS Management Data Access Service shall have the capability to schedule the transfer of management data at the sites to the SMC.
	C-MSS-18040	The MSS Management Data Access Service shall maintain the integrity of the management database.
	C-MSS-18060	The MSS Management Data Access Service shall provide the capability for an application to access management data.
TC030.004	C-MSS-18340	The MSS Management Data Access Service shall provide the capability for an application to selectively read a record from a log file.
	C-MSS-18350	The MSS Management Data Access Service shall provide the capability for an application to load log files into the management database at the site
	C-MSS-18260	The MSS Management Data Access Service shall have the capability to schedule the transfer and loading log files into the management database at the site.
	C-MSS-18220	The MSS Management Data Access Service shall provide the capability for an application via APIs to alter tables and fields in the management database.
	C-MSS-18270	The MSS Management Data Access Service shall have the capability to schedule the archiving of log files at the site.
	C-MSS-18280	The MSS Management Data Access Service shall have the capability to schedule the transfer of management data at the sites to the SMC.
	C-MSS-18040	The MSS Management Data Access Service shall maintain the integrity of the management database.

Table A-1. Requirements Verification Matrix (68 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-18060	The MSS Management Data Access Service shall provide the capability for an application to access management data.
TC030.005	C-MSS-66135	The MSS Performance Management Application Service shall have the capability to calculate the following statistics for the purpose of supporting RMA analysis for managed objects: a. Mean Down time (MDT) b. Mean Time Between Maintenance (MTBM) 1. Mean Time Between Preventive Maintenance (MTBPM) 2. Mean Time Between Corrective Maintenance (MTBCM) c. Mean Time to Repair (MTTR)
	C-MSS-66137	The MSS Performance Management Application Service shall retain the calculated RMA statistics in a repository accessible for further analysis by the M&O Staff.
	C-MSS-18040	The MSS Management Data Access Service shall maintain the integrity of the management database.
	C-MSS-18060	The MSS Management Data Access Service shall provide the capability for an application to access management data.
	C-MSS-18340	The MSS Management Data Access Service shall provide the capability for an application to selectively read a record from a log file.
	C-MSS-18260	The MSS Management Data Access Service shall have the capability to schedule the transfer and loading log files into the management database at the site.
	C-MSS-18070	The MSS Management Data Access Service shall provide the capability to selectively access management data.
TC030.006	C-MSS-66135	The MSS Performance Management Application Service shall have the capability to calculate the following statistics for the purpose of supporting RMA analysis for managed objects: a. Mean Down time (MDT) b. Mean Time Between Maintenance (MTBM) 1. Mean Time Between Preventive Maintenance (MTBPM) 2. Mean Time Between Corrective Maintenance (MTBCM) c. Mean Time to Repair (MTTR)
	C-MSS-66137	The MSS Performance Management Application Service shall retain the calculated RMA statistics in a repository accessible for further analysis by the M&O Staff.
	C-MSS-18040	The MSS Management Data Access Service shall provide the capability for an application to selectively read a record from a log file.
	C-MSS-18060	The MSS Management Data Access Service shall have the capability to schedule the transfer and loading log files into the management database at the site.
	C-MSS-18340	The MSS Management Data Access Service shall provide the capability to selectively access management data.

Table A-1. Requirements Verification Matrix (69 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-18260	The MSS Management Data Access Service shall have the capability to schedule the transfer and loading log files into the management database at the site.
	C-MSS-18070	The MSS Management Data Access Service shall provide the capability to selectively access management data.
TC031.001	C-MSS-40000	The MSS configuration management application service at each site shall track the following items at the site by name and identifier: a. ECS subsystems, networks, and configured system and network devices such as workstations, servers, and routers b. ECS releases and site baselines c. ECS hardware and software resources designated as configuration items d. specifications asscoiated with configuration items e. technical documentation and test materials f. scientific algorithms, including software, data and test materials (DAACs only)
	C-MSS-40100	The MSS configuration management application service at the SMC and the DAACs shall maintain SCF-provided configuration data for individual algorithms, including: a. algorithm development version numbers, identification codes, and reference numbers; b. SCF point of contact's name and organization; c. associated files' names, formats, sizes, and descriptions; d. number of files by category and type.
	C-MSS-40120	The MSS configuration management application service at the SMC shall track the names and identifiers of the following items deployed at the sites: a. ECS subsystems, networks, and configured system and network devices such as workstations, servers, and routers b. ECS releases and baselines c. ECS hardware and software resources designated as configuration items d. technical documentation and test materials; e. scientific algorithms, including software, data and test materials (DAAC's only) f. algorithm processing logic control and calibration coefficients data; g. algorithm test documentation, including specifications, data files, and scripts.
	C-MSS-40240	The MSS configuration management application service at the SMC shall maintain software-critical and security-sensitive items lists.

Table A-1. Requirements Verification Matrix (70 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-2510	The SMC shall provide at a minimum system-wide configuration management for the operational hardware, scientific and system software, and the SMC toolkit contained within ECS. The management system shall support the migration of hardware and software upgrades into the operational environment.
	SMC-2515	The LSM shall provide configuration management for at least the operational hardware, system software, and scientific software within its element and for the migration of enhancements into the operational system.
	SMC-6340	The SMC shall track system configuration that, at a minimum, audits: a. Hardware resources b. Software resources
	PGS-0950	The PGS shall interface to maintain configuration control of all algorithms and calibration coefficients used in operational Standard Product production. Controlled information shall contain at a minimum: a. Source code including version number and author b. Benchmark test procedures, test data, and results c. Date and time of operational installation d. Compiler identification and version e. Final algorithm documentation
TC031.002	C-MSS-40100	The MSS configuration management application service at the SMC and the DAACs shall maintain SCF-provided configuration data for individual algorithms, including: a. algorithm development version numbers, identification codes, and reference numbers; b. SCF point of contact's name and organization; c. associated files' names, formats, sizes, and descriptions; d. number of files by category and type.
	SMC-6345	The LSM shall, as needed, perform configuration accountability to include, at a minimum, the audit of hardware and software resources within its element.
TC031.003	C-MSS-40030	The MSS configuration management application service at each site shall make available to the SMC records that identify the site's operational baseline and the versions and implementation status of configuration controlled resources that comprise it.
	C-MSS-40040	The MSS configuration management application service at each site shall make available to the SMC, "level of assembly" records that describe the composition of configuration items at the site.
	C-MSS-40280	The MSS configuration management application service shall characterize ECS-controlled resources as system-wide or site-specific.

Table A-1. Requirements Verification Matrix (71 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-2515	The LSM shall provide configuration management for at least the operational hardware, system software, and scientific software within its element and for the migration of enhancements into the operational system.
TC031.004	C-MSS-40060	The MSS configuration management application service at each site shall maintain historical status records about ECS configuration items at the site, identifying each item's: a. current version; b. current version's specifications and technical, operations, and maintenance documentation; c. specification and technical documentation history; d. "level of assembly" representation of the components comprising the items current and release configurations e. version history
	C-MSS-40070	The MSS configuration management application service at the SMC and the sites shall maintain records that establish traceability among operational baselines and releases.
	C-MSS-40080	The MSS configuration management application service at the SMC and the sites shall maintain records describing dependencies among baseline objects.
	C-MSS-40110	The MSS configuration management application service shall display and report indentured, "level of assembly" lists that describe the component structure of configuration items.
	C-MSS-40200	The MSS configuration management application service at the SMC shall maintain historical status records about ECS configuration items system-wide, to include each item's: a. current version; b. current version's specifications and technical, operations, and maintenance documentation; c. specifications and technical documentation history d. "level of assembly" representation of components comprising the item's current and release configurations: e. version history
	C-MSS-40210	The MSS configuration management application service at the SMC shall maintain historical status records about ECS system releases, to include each release's: a. latest baseline plus approved changes. b. baseline history. c. latest release documentation. d. "level of assembly" representation of the subsystem and configuration item versions that comprise the release configuration e. history of changes, including changes to subordinate units/components. f. effectivity and installation status at operational sites. g. release configuration

Table A-1. Requirements Verification Matrix (72 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-2510	The SMC shall provide at a minimum system-wide configuration management for the operational hardware, scientific and system software, and the SMC toolkit contained within ECS. The management system shall support the migration of hardware and software upgrades into the operational environment.
	SMC-2515	The LSM shall provide configuration management for at least the operational hardware, system software, and scientific software within its element and for the migration of enhancements into the operational system.
TC031.005	C-MSS-40140	The MSS configuration management application service at the SMC shall maintain, and make available system-wide, information identifying the sites where individual versions of configuration items are located and the operational status of that version at the site.
	C-MSS-40150	The MSS configuration management application service at the SMC shall maintain, and make available system-wide, records that identify the current and previous versions of ECS hardware and software resources deployed to the sites.
	C-MSS-40160	The MSS configuration management application service at the SMC shall maintain records that identify the current and previous versions of ECS documents associated with deployed ECS resources.
	C-MSS-40180	The MSS configuration management application service at the SMC shall maintain, and distribute to each site, records that identify the specifications and technical, operations, and maintenance documents associated with versions of ECS hardware and software configuration items deployed to the site.
	C-MSS-40250	The MSS configuration management application service at the SMC shall produce, and make available system-wide, reports containing the identity and change status of documents associated with deployed ECS resources.
	C-MSS-40260	The MSS configuration management application service at the SMC shall produce, and make available system-wide, reports, containing the identity and change status of individual ECS resources deployed to the sites.
	C-MSS-40270	The MSS configuration management application service at the SMC shall produce, and make available system-wide, reports containing the identity of resources comprising ECS baselines and releases.
	SMC-2510	The SMC shall provide at a minimum system-wide configuration management for the operational hardware, scientific and system software, and the SMC toolkit contained within ECS. The management system shall support the migration of hardware and software upgrades into the operational environment.

Table A-1. Requirements Verification Matrix (73 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-2515	The LSM shall provide configuration management for at least the operational hardware, system software, and scientific software within its element and for the migration of enhancements into the operational system.
	SMC-6340	The SMC shall track system configuration that, at a minimum, audits: a. Hardware resources b. Software resources
TC031.006	C-MSS-40170	The MSS configuration management application service at the SMC shall maintain, and distribute to each site, records that identify the baseline changes included in each release of ECS hardware and software deployed to the site.
	C-MSS-40190	MSS configuration management application service at the SMC shall maintain, and distribute to each site, records that describe the change requests (enhancements and corrections) satisfied by new versions of ECS hardware, software, and documentation deployed to the sites.
	C-MSS-40220	The MSS configuration management application service at the SMC shall maintain historical status records about ECS baseline changes to include: a. sites affected; b. installation dates c. installation status.
	SMC-2510	The SMC shall provide at a minimum system-wide configuration management for the operational hardware, scientific and system software, and the SMC toolkit contained within ECS. The management system shall support the migration of hardware and software upgrades into the operational environment.
TC031.007	C-MSS-40290	The MSS configuration management application service shall accept and store baseline management data records provided via interactive user interface and formatted data files.
TC031.008	C-MSS-40300	The MSS configuration management application service shall produce formatted data files containing baseline management data records.
TC031.009	C-MSS-40990	The MSS configuration management application service shall log the following information for configuration management events: operation type; userid of initiator; date-time stamp; host name.

Table A-1. Requirements Verification Matrix (74 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-40995	The MSS configuration management application service shall generate chronological reports of logged CM events associated with M&O staff-selectable: time frames; operation types; userids; hosts.
BC009.001	DADS0901#A	The DADS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management h. Distribution and Ingest Management
	PGS-0310#A	The PGS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management c. Accounting Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management.
	EOSD0510#A	ECS shall be capable of being tested during all phases of its development and flight operations.
	EOSD0740#A	Each ECS element shall provide a set of real or simulated functional capabilities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. ECS System (Integration of ECS elements)
	EOSD0750#A	Each ECS element shall provide a set of real or simulated functions which interfaces with both its ECS internal and external entities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. EOSDIS System (Integration of EOSDIS elements)
	EOSD0760#A	Each ECS element shall support end-to-end EOS system testing and fault isolation.

Table A-1. Requirements Verification Matrix (75 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD0500#A	ECS shall perform the following major functions: a. EOS Mission Planning and Scheduling b. EOS Mission Operations c. Command and Control d. Communications and Networking e. Data Input f. Data Processing g. Data Storage h. Data Distribution i. Information Management j. End-to-End Fault Management k. System Management
	EOSD1703#A	ECS shall provide maintenance and operations interfaces to the DAACs to support the functions of: a). System Management b). Science Algorithm Integration c). Product Generation d). Data Archive/Distribution e). User Support Services f). System Maintenance
	NI-0460#A	ECS shall have the capability to receive periodic information regarding NOLAN network performance and link utilization.
	SMC-1345#A	The LSM shall perform priority management services to resolve conflicts for ECS resources.
	SMC-3345#A	The LSM shall perform quality assurance for its site/elements performance as well as programmatic areas that includes, at a minimum: a. Quality testing, benchmarks and audits for element enhancement implementations b. Quality checking and audits of products processed and delivered c. Quality testing and audits of element resource performance,
	SMC-3355#A	The LSM shall implement the performance criteria from SMC (including parametric limits and operational threshold levels) for evaluating element resource performance.
	SMC-3375#A	For each limit checked parameter, the LSM (including those thresholds directed by the SMC) shall have the capability of evaluating multiple levels of thresholds including, at a minimum: a. On/off b. Pass/fail c. Various levels of degradation
	SMC-3385#A	The LSM shall evaluate system performance against the ESDIS project established performance criteria.
	SMC-3395#A	The LSM shall generate, in response to each limit check threshold, alert indicators of fault or degraded conditions.

Table A-1. Requirements Verification Matrix (76 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-3397#A	The LSM shall generate, as needed, requests for performance testing, including, at a minimum: a. Resource to be tested b. Test purpose c. Requested test priority d. Required test environment e. Impacts to operations f. Expected test results
	SMC-3415#A	The LSM shall perform short and long-term trend analysis of element performance, including, at a minimum: a. Operational status b. Performance of a particular resource c. Maintenance activities (e.g., number of repairs per item)
	EOSD0545#A	ECS shall be able to accommodate growth (e.g., capacity) in all of its functions as well as the addition of new functions.
	EOSD3490#A	Reliability statistics for ECS shall be collected and monitored using the Mean Time Between Maintenance (MTBM) for each component and operational capability.
	EOSD3492#A	RMA data shall be maintained in a repository accessible for logistics analysis and other purposes.
	EOSD3620#A	ECS shall predict and periodically assess maintainability by measuring the actual MDT and comparing to the required MDT.
	EOSD3630#A	The maximum down time shall not exceed twice the required MDT in 99 percent of failure occurrences.
	EOSD3700#A	ECS functions shall have an operational availability of 0.96 at a minimum (.998 design goal) and an MDT of four (4) hours or less (1.5 hour design goal), unless otherwise specified. The above requirement covers equipment including: a. "Non-critical" equipment configured with the critical equipment supporting the functional capabilities in the requirements. b. Equipment providing other functionality not explicitly stated in the RMA requirements that follow.
	EOSD3710#A	The ECS shall have no single point of failure for functions associated with real-time operations of the spacecraft and instruments.
	EOSD4035#A	The ESN shall have no single point of failure for functions associated with network databases and configuration data.
	EOSD4036#A	The operational availability of individual ESN segments shall be consistent with the specified operational availability of the supported ECS functions.

Table A-1. Requirements Verification Matrix (77 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD5000#A	ECS shall enable the addition of other data providers, e.g. DAACs, SCFs, ADCs, ODCs, which may: - provide heterogeneous services, i.e. services in support of EOS which may be less than or different than ECS services. - be connected with varying topologies - have variable levels of reliability or operational availability.
	EOSD5070#A	ECS shall enable expansion to GByte networks including the ability to provide increased volume of data distribution/access.
	ESN-0815#A	Network simulation and traffic modeling capability shall be provided to troubleshoot network problems and to use in network planning.
	ESN-1060#A	The ESN performance management function shall provide the capability to evaluate the performance of ESN resources and interconnection activities.
	ESN-1065#A	The ESN performance management function shall include trend analysis for prediction of loading and bottlenecks/delays.
	ESN-1090#A	The ESN shall provide the capability to control the communications performance parameters of the network.
	ESN-1206#A	The ESN capacity and performance shall be consistent with the specified capacity and performance requirements of the ECS functions.
	ESN-1207#A	The ESN capacity and performance shall be capable of expansion to be consistent with the specified capacity and performance growth requirements of the ECS elements and functions.
	ESN-0240#A	The ESN shall be extensible in its design to provide capability for growth and enhancement.
	ESN-0740#A	The ESN network management service shall retrieve performance/fault data about ESN protocol stacks and equipment.
	ESN-0750#A	The ESN shall provide statistical processing capabilities to allow extraction and tabulation of network performance data.
	ESN-0780#A	The network elements including the Internet interfaces, shall have the capability to report, periodically and on an interactive basis, network statistics to the ESN network management function, including the following information: a. Network round trip delay b. Network reset and restart indications c. Outages and CRC errors d. Performance statistics
	EOSD3510#A	Reliability predictions shall be calculated in accordance with the parts count analysis method, Appendix A, of MIL-HDBK-217F, Reliability Prediction of Electronic Equipment.
	EOSD3600#A	Maintainability shall be predicted in accordance with MIL-HDBK-472, Maintainability Prediction, Procedure IV.

Table A-1. Requirements Verification Matrix (78 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD3625#A	For ECS functions with a backup capability, ECS shall use switchover time to the backup capability in measuring maintainability, rather than down time, when the component goes down.
	C-MSS-90280	The DBMS shall provide the capability to issue and record a database checkpoint.
	C-MSS-90260	The DBMS shall provide a capability to export, archival, and restore a database.
	C-MSS-90240	The DBMS shall provide for automatic database recovery including a means to: a. automatically restore undamaged portions of a database and recover work in progress after a system or component failure b. achieve dynamic backout of database modifications, performed by a failing transaction, that does not affect separate, concurrent tasks
	C-MSS-90230	The DBMS shall provide a transaction roll backward capability to a specified time or state: a. Restore a database b. Restore all or operator selected database objects of any database
	C-MSS-90210	The DBMS shall support the following features: a. Data compression of nulls and variable length character strings, and indexes b. Space reclaimed from deleted records automatically c. Variable-length column storage
	C-MSS-90200	The DBMS shall perform on-line disk management functions to include: a. Relocation of database files to different disks b. Expansion of database size by adding new physical data files to it on-line c. Dynamic pre-allocation of contiguous space for tables d. Database objects and indexes can span physical files e. Database objects and indexes can exist on different disks
	C-MSS-90190	The DBMS shall provide capabilities for specifying frequency, time, and type of backups.
	C-MSS-90150	The DBMS shall support access structures (i.e., single-level indexes, multilevel indexes) to improve the efficiency of retrieval of management data.
	C-MSS-90140	The DBMS shall support, or be accessed via, CSS session-establishment services.
	C-MSS-90520	The Report Generator shall have the capability to generate ad hoc reports from management data maintained in the DBMS.

Table A-1. Requirements Verification Matrix (79 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-90530	The Report Generator shall provide the capability to format reports to include the report: a._title b._header c._footer d._page number e._date/time of report
	C-MSS-90500	The Report Generator shall be compatible with the DBMS.
	C-MSS-90570	The Report Generator shall have the capability to generate charts and graphs (e.g., bar, pie, line, etc.) from management data maintained in the DBMS.
	C-MSS-90600	The Report Generator shall provide the capability to redirect generated reports to: a._console b._disk file c._printer
	C-MSS-91010	The MSS Office Automation word processing capability shall facilitate the: a._preparation, revision, and recording of documents, messages, reports, and data b._import, transformation, and editing of documents produced by other word processing packages c._insertion of worksheet and graphic images into documents, messages, and reports d._transfer of document, message, and report information to spreadsheet and graphics applications e._printing of documents, messages, reports, and data
	C-MSS-90510	The Report Generator shall provide a Motif based Graphical User Interface (GUI) for creating ad hoc reports.
BC009.002	DADS0901#A	The DADS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management h. Distribution and Ingest Management

Table A-1. Requirements Verification Matrix (80 of 134)

Test Case ID	Requirement ID	Requirement Text
	PGS-0310#A	The PGS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management c. Accounting Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management.
	EOSD0510#A	ECS shall be capable of being tested during all phases of its development and flight operations.
	EOSD0740#A	Each ECS element shall provide a set of real or simulated functional capabilities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. ECS System (Integration of ECS elements)
	EOSD0750#A	Each ECS element shall provide a set of real or simulated functions which interfaces with both its ECS internal and external entities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. EOSDIS System (Integration of EOSDIS elements)
	EOSD0760#A	Each ECS element shall support end-to-end EOS system testing and fault isolation.
	EOSD0500#A	ECS shall perform the following major functions: a. EOS Mission Planning and Scheduling b. EOS Mission Operations c. Command and Control d. Communications and Networking e. Data Input f. Data Processing g. Data Storage h. Data Distribution i. Information Management j. End-to-End Fault Management k. System Management
	EOSD1703#A	ECS shall provide maintenance and operations interfaces to the DAACs to support the functions of: a). System Management b). Science Algorithm Integration c). Product Generation d). Data Archive/Distribution e). User Support Services f). System Maintenance

Table A-1. Requirements Verification Matrix (81 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-1345#A	The LSM shall perform priority management services to resolve conflicts for ECS resources.
	SMC-3345#A	The LSM shall perform quality assurance for its site/elements performance as well as programmatic areas that includes, at a minimum: a. Quality testing, benchmarks and audits for element enhancement implementations b. Quality checking and audits of products processed and delivered c. Quality testing and audits of element resource performance,
	SMC-3355#A	The LSM shall implement the performance criteria from SMC (including parametric limits and operational threshold levels) for evaluating element resource performance.
	SMC-3375#A	For each limit checked parameter, the LSM (including those thresholds directed by the SMC) shall have the capability of evaluating multiple levels of thresholds including, at a minimum: a. On/off b. Pass/fail c. Various levels of degradation
	SMC-3385#A	The LSM shall evaluate system performance against the ESDIS project established performance criteria.
	SMC-3395#A	The LSM shall generate, in response to each limit check threshold, alert indicators of fault or degraded conditions.
	SMC-3397#A	The LSM shall generate, as needed, requests for performance testing, including, at a minimum: a. Resource to be tested b. Test purpose c. Requested test priority d. Required test environment e. Impacts to operations f. Expected test results
	SMC-3415#A	The LSM shall perform short and long-term trend analysis of element performance, including, at a minimum: a. Operational status b. Performance of a particular resource c. Maintenance activities (e.g., number of repairs per item)
	SMC-4325#A	The LSM shall request fault diagnosis testing be performed, including, at a minimum: a. Software and hardware tolerance testing b. Resource-to-resource connectivity testing within its element

Table A-1. Requirements Verification Matrix (82 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-5305#A	The LSM shall maintain security policies and procedures, including, at a minimum: a. Physical security b. Password management c. Operational security d. Data classifications e. Access/privileges f. Compromise mitigation
	SMC-5325#A	The LSM shall promulgate, maintain, authenticate, and monitor user and device accesses and privileges.
	SMC-5335#A	The LSM shall perform security testing that includes, at a minimum, password auditing and element internal access/privileges checking.
	SMC-5345#A	The LSM shall perform compromise (e.g., virus or worm penetration) risk analysis, and detection.
	SMC-5355#A	The LSM shall isolate the compromised area, detach the compromised input I/O, and the compromised areas output I/O until the compromise has been eliminated.
	SMC-5365#A	The LSM shall generate recovery actions in response to the detection of compromises.
	SMC-6315#A	The LSM shall perform, as needed, security audit trails within its element.
	EOSD0545#A	ECS shall be able to accommodate growth (e.g., capacity) in all of its functions as well as the addition of new functions.
	EOSD0040#A	ECS shall provide users without prior approved accounts access to the system for descriptive information about ECS and the types of data it contains.
	EOSD1990#A	The ECS system and elements shall employ security measures and techniques for all applicable security disciplines which are identified in the preceding documents. These documents shall provide the basis for the ECS security policy.
	EOSD2400#A	ECS shall provide multiple categories of data protection based on the sensitivity levels of ECS data, as defined in NHB 2410.9.
	EOSD2430#A	Data base access and manipulation shall accommodate control of user access and update of security controlled data.
	EOSD2480#A	ECS elements shall require unique sessions when security controlled data are being manipulated.

Table A-1. Requirements Verification Matrix (83 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD2510#A	ECS elements shall maintain an audit trail of: a. All accesses to the element security controlled data b. Users/processes/elements requesting access to element security controlled data c. Data access/manipulation operations performed on security controlled data d. Date and time of access to security controlled data e. Unsuccessful access attempt to the element security controlled data by unauthorized users/elements/processes f. Detected computer system viruses and worms g. Actions taken to contain or destroy a virus
	EOSD2550#A	The ECS elements shall limit use of master passwords or use of a single password for large organizations requiring access to a mix of security controlled and non-sensitive data.
	EOSD2620#A	ECS elements shall disconnect a user/element after a predetermined number of unsuccessful attempts to access data.
	EOSD2640#A	ECS elements shall relinquish a connection between the element and a user when the user has not been active for a configurable period of time.
	EOSD2650#A	ECS elements shall report detected security violations to the SMC.
	EOSD2660#A	ECS elements shall at all times maintain and comply with the security directives issued by the SMC.
	EOSD2710#A	ECS elements shall report all detected computer viruses and actions taken to the SMC.
	EOSD3000#A	The ECS shall provide for security safeguards to cover unscheduled system shutdown (aborts) and subsequent restarts, as well as for scheduled system shutdown and operational startup.
	EOSD3200#A	A minimum of one backup which is maintained in a separate physical location (i.e., different building) shall be maintained for ECS software and key data items (including security audit trails and logs).
	ESN-1360#A	The ESN shall control access of processes and users through an authentication and authorization service that meets GNMP standards.
	ESN-1365#A	The ESN shall isolate FOS with secure interfaces.
	ESN-1380#A	The ESN shall provide countermeasures for the following security threats related to data communications: a. modification of data (i.e., manipulation) while in transit over the network b. disclosure of authentication information c. degradation in network or processing resource performance through denial of service attack d. Impersonation of authentication credentials or authorization privileges.

Table A-1. Requirements Verification Matrix (84 of 134)

Test Case ID	Requirement ID	Requirement Text
	ESN-1400#A	The following security functions and services, at a minimum, shall be provided: a. authentication b. access (authorization) control c. data integrity d. data confidentiality
	ESN-1430#A	The ESN shall provide the following security event functions: a. Event detection b. Event reporting c. Event logging
	EOSD2100#A	The ECS technical security policy planning shall be comprehensive and shall cover at least the following areas: a. Applicability of the C2 Level of Trustedness as defined by the NSA b. Applicability of the C2 Object Reuse capability c. Discretionary control and monitoring of user access d. ECS communications, network access, control, and monitoring e. Computer system "virus" monitoring, detection, and remedy f. Data protection controls g. Account/privilege management and user session tailoring h. Restart/recovery i. Security audit trail generation j. Security analysis and reporting k. Risk analysis
	EOSD2200#A	Selection criteria meeting overall ECS security policies and system requirements shall be applied when selecting hardware.
	ESN-0740#A	The ESN network management service shall retrieve performance/fault data about ESN protocol stacks and equipment.
	ESN-0750#A	The ESN shall provide statistical processing capabilities to allow extraction and tabulation of network performance data.
	ESN-0780#A	The network elements including the Internet interfaces, shall have the capability to report, periodically and on an interactive basis, network statistics to the ESN network management function, including the following information: a. Network round trip delay b. Network reset and restart indications c. Outages and CRC errors d. Performance statistics
	EOSD3510#A	Reliability predictions shall be calculated in accordance with the parts count analysis method, Appendix A, of MIL-HDBK-217F, Reliability Prediction of Electronic Equipment.
	EOSD3600#A	Maintainability shall be predicted in accordance with MIL-HDBK-472, Maintainability Prediction, Procedure IV.

Table A-1. Requirements Verification Matrix (85 of 134)

Test Case ID	Requirement ID	Requirement Text
	NI-0470#A	ECS shall have the capability to receive notifications of security breaches at NOLAN sites or within the NOLAN network that could potentially affect ECS sites.
	NI-0480#A	ECS shall have the capability to send to NOLAN notifications of security breaches at ECS facilities that could affect NOLAN and other EOSDIS sites.
BC009.003	DADS0901#A	The DADS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management h. Distribution and Ingest Management
	PGS-0310#A	The PGS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management c. Accounting Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management.
	EOSD0510#A	ECS shall be capable of being tested during all phases of its development and flight operations.
	EOSD0740#A	Each ECS element shall provide a set of real or simulated functional capabilities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. ECS System (Integration of ECS elements)
	EOSD0750#A	Each ECS element shall provide a set of real or simulated functions which interfaces with both its ECS internal and external entities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. EOSDIS System (Integration of EOSDIS elements)
	EOSD0760#A	Each ECS element shall support end-to-end EOS system testing and fault isolation.

Table A-1. Requirements Verification Matrix (86 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD0500#A	ECS shall perform the following major functions: a. EOS Mission Planning and Scheduling b. EOS Mission Operations c. Command and Control d. Communications and Networking e. Data Input f. Data Processing g. Data Storage h. Data Distribution i. Information Management j. End-to-End Fault Management k. System Management
	EOSD1703#A	ECS shall provide maintenance and operations interfaces to the DAACs to support the functions of: a). System Management b). Science Algorithm Integration c). Product Generation d). Data Archive/Distribution e). User Support Services f). System Maintenance
	SMC-1345#A	The LSM shall perform priority management services to resolve conflicts for ECS resources.
	SMC-3345#A	The LSM shall perform quality assurance for its site/elements performance as well as programmatic areas that includes, at a minimum: a. Quality testing, benchmarks and audits for element enhancement implementations b. Quality checking and audits of products processed and delivered c. Quality testing and audits of element resource performance,
	SMC-3355#A	The LSM shall implement the performance criteria from SMC (including parametric limits and operational threshold levels) for evaluating element resource performance.
	SMC-3375#A	For each limit checked parameter, the LSM (including those thresholds directed by the SMC) shall have the capability of evaluating multiple levels of thresholds including, at a minimum: a. On/off b. Pass/fail c. Various levels of degradation
	SMC-3385#A	The LSM shall evaluate system performance against the ESDIS project established performance criteria.
	SMC-3395#A	The LSM shall generate, in response to each limit check threshold, alert indicators of fault or degraded conditions.

Table A-1. Requirements Verification Matrix (87 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-3397#A	The LSM shall generate, as needed, requests for performance testing, including, at a minimum: a. Resource to be tested b. Test purpose c. Requested test priority d. Required test environment e. Impacts to operations f. Expected test results
	SMC-3415#A	The LSM shall perform short and long-term trend analysis of element performance, including, at a minimum: a. Operational status b. Performance of a particular resource c. Maintenance activities (e.g., number of repairs per item)
	EOSD0545#A	ECS shall be able to accommodate growth (e.g., capacity) in all of its functions as well as the addition of new functions.
	EOSD3490#A	Reliability statistics for ECS shall be collected and monitored using the Mean Time Between Maintenance (MTBM) for each component and operational capability.
	EOSD3492#A	RMA data shall be maintained in a repository accessible for logistics analysis and other purposes.
	EOSD3620#A	ECS shall predict and periodically assess maintainability by measuring the actual MDT and comparing to the required MDT.
	EOSD3630#A	The maximum down time shall not exceed twice the required MDT in 99 percent of failure occurrences.
	EOSD3700#A	ECS functions shall have an operational availability of 0.96 at a minimum (.998 design goal) and an MDT of four (4) hours or less (1.5 hour design goal), unless otherwise specified. The above requirement covers equipment including: a. "Non-critical" equipment configured with the critical equipment supporting the functional capabilities in the requirements. b. Equipment providing other functionality not explicitly stated in the RMA requirements that follow.
	EOSD3710#A	The ECS shall have no single point of failure for functions associated with real-time operations of the spacecraft and instruments.
	EOSD4035#A	The ESN shall have no single point of failure for functions associated with network databases and configuration data.
	EOSD4036#A	The operational availability of individual ESN segments shall be consistent with the specified operational availability of the supported ECS functions.

Table A-1. Requirements Verification Matrix (88 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD5000#A	ECS shall enable the addition of other data providers, e.g. DAACs, SCFs, ADCs, ODCs, which may: - provide heterogeneous services, i.e. services in support of EOS which may be less than or different than ECS services. - be connected with varying topologies - have variable levels of reliability or operational availability.
	EOSD5070#A	ECS shall enable expansion to GByte networks including the ability to provide increased volume of data distribution/access.
	ESN-1000#A	The ESN network management function shall have the capability to build histories for different types of errors and events, and the capability to analyze errors and recommend corrective action wherever practical.
	ESN-1070#A	The ESN shall provide the capability to perform the following functions, at a minimum: a. generate/collect network statistics b. control collection/generation of network statistics c. store system statistics and statistical histories d. display the system statistics e. track end-to-end transaction performance
	ESN-0280#A	The ESN shall provide file transfer and management service and as a minimum shall include the capability to transfer the following data types: a. Unstructured Text b. Binary Unstructured c. Binary Sequential d. Sequential Text
	ESN-0290#A	The file transfer and management service shall be available in interactive and non-interactive services.
	ESN-0300#A	The file transfer and management non-interactive services shall be able to be scheduled.
	EOSD5110#A	ECS shall enable the separate use of data management, data processing, or data archive and distribution software components by a GCDIS data center. The GCDIS data centers will have full responsibility for integration of those components within their environment. Interfaces between the components must be developed to serve the mission of EOSDIS, but be made available for a GCDIS data center.
	ESN-0815#A	Network simulation and traffic modeling capability shall be provided to troubleshoot network problems and to use in network planning.
	ESN-0830#A	The ESN shall have the capability to detect and report communications related errors and events both locally and at the ESN network management facility.

Table A-1. Requirements Verification Matrix (89 of 134)

Test Case ID	Requirement ID	Requirement Text
	ESN-1060#A	The ESN performance management function shall provide the capability to evaluate the performance of ESN resources and interconnection activities.
	ESN-1065#A	The ESN performance management function shall include trend analysis for prediction of loading and bottlenecks/delays.
	ESN-1090#A	The ESN shall provide the capability to control the communications performance parameters of the network.
	ESN-1206#A	The ESN capacity and performance shall be consistent with the specified capacity and performance requirements of the ECS functions.
	ESN-1207#A	The ESN capacity and performance shall be capable of expansion to be consistent with the specified capacity and performance growth requirements of the ECS elements and functions.
	ESN-0240#A	The ESN shall be extensible in its design to provide capability for growth and enhancement.
	ESN-0740#A	The ESN network management service shall retrieve performance/fault data about ESN protocol stacks and equipment.
	ESN-0750#A	The ESN shall provide statistical processing capabilities to allow extraction and tabulation of network performance data.
	ESN-0780#A	The network elements including the Internet interfaces, shall have the capability to report, periodically and on an interactive basis, network statistics to the ESN network management function, including the following information: a. Network round trip delay b. Network reset and restart indications c. Outages and CRC errors d. Performance statistics
	EOSD3510#A	Reliability predictions shall be calculated in accordance with the parts count analysis method, Appendix A, of MIL-HDBK-217F, Reliability Prediction of Electronic Equipment.
	EOSD3600#A	Maintainability shall be predicted in accordance with MIL-HDBK-472, Maintainability Prediction, Procedure IV.
	EOSD3625#A	For ECS functions with a backup capability, ECS shall use switchover time to the backup capability in measuring maintainability, rather than down time, when the component goes down.

Table A-1. Requirements Verification Matrix (90 of 134)

Test Case ID	Requirement ID	Requirement Text
BC009.004	DADS0901#A	The DADS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management h. Distribution and Ingest Management
	PGS-0310#A	The PGS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management c. Accounting Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management.
	EOSD0510#A	ECS shall be capable of being tested during all phases of its development and flight operations.
	EOSD0740#A	Each ECS element shall provide a set of real or simulated functional capabilities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. ECS System (Integration of ECS elements)
	EOSD0750#A	Each ECS element shall provide a set of real or simulated functions which interfaces with both its ECS internal and external entities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. EOSDIS System (Integration of EOSDIS elements)
	EOSD0760#A	Each ECS element shall support end-to-end EOS system testing and fault isolation.
	EOSD0500#A	ECS shall perform the following major functions: a. EOS Mission Planning and Scheduling b. EOS Mission Operations c. Command and Control d. Communications and Networking e. Data Input f. Data Processing g. Data Storage h. Data Distribution i. Information Management j. End-to-End Fault Management k. System Management

Table A-1. Requirements Verification Matrix (91 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD1703#A	ECS shall provide maintenance and operations interfaces to the DAACs to support the functions of: a). System Management b). Science Algorithm Integration c). Product Generation d). Data Archive/Distribution e). User Support Services f). System Maintenance
	SMC-1345#A	The LSM shall perform priority management services to resolve conflicts for ECS resources.
	SMC-3345#A	The LSM shall perform quality assurance for its site/elements performance as well as programmatic areas that includes, at a minimum: a. Quality testing, benchmarks and audits for element enhancement implementations b. Quality checking and audits of products processed and delivered c. Quality testing and audits of element resource performance,
	SMC-3355#A	The LSM shall implement the performance criteria from SMC (including parametric limits and operational threshold levels) for evaluating element resource performance.
	SMC-3375#A	For each limit checked parameter, the LSM (including those thresholds directed by the SMC) shall have the capability of evaluating multiple levels of thresholds including, at a minimum: a. On/off b. Pass/fail c. Various levels of degradation
	SMC-3385#A	The LSM shall evaluate system performance against the ESDIS project established performance criteria.
	SMC-3395#A	The LSM shall generate, in response to each limit check threshold, alert indicators of fault or degraded conditions.
	SMC-3397#A	The LSM shall generate, as needed, requests for performance testing, including, at a minimum: a. Resource to be tested b. Test purpose c. Requested test priority d. Required test environment e. Impacts to operations f. Expected test results
	SMC-3415#A	The LSM shall perform short and long-term trend analysis of element performance, including, at a minimum: a. Operational status b. Performance of a particular resource c. Maintenance activities (e.g., number of repairs per item)

Table A-1. Requirements Verification Matrix (92 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD0545#A	ECS shall be able to accommodate growth (e.g., capacity) in all of its functions as well as the addition of new functions.
	EOSD3490#A	Reliability statistics for ECS shall be collected and monitored using the Mean Time Between Maintenance (MTBM) for each component and operational capability.
	EOSD3492#A	RMA data shall be maintained in a repository accessible for logistics analysis and other purposes.
	EOSD3620#A	ECS shall predict and periodically assess maintainability by measuring the actual MDT and comparing to the required MDT.
	EOSD3630#A	The maximum down time shall not exceed twice the required MDT in 99 percent of failure occurrences.
	EOSD3700#A	ECS functions shall have an operational availability of 0.96 at a minimum (.998 design goal) and an MDT of four (4) hours or less (1.5 hour design goal), unless otherwise specified. The above requirement covers equipment including: a. "Non-critical" equipment configured with the critical equipment supporting the functional capabilities in the requirements. b. Equipment providing other functionality not explicitly stated in the RMA requirements that follow.
	EOSD3710#A	The ECS shall have no single point of failure for functions associated with real-time operations of the spacecraft and instruments.
	EOSD4035#A	The ESN shall have no single point of failure for functions associated with network databases and configuration data.
	EOSD4036#A	The operational availability of individual ESN segments shall be consistent with the specified operational availability of the supported ECS functions.
	EOSD5000#A	ECS shall enable the addition of other data providers, e.g. DAACs, SCFs, ADCs, ODCs, which may: - provide heterogeneous services, i.e. services in support of EOS which may be less than or different than ECS services. - be connected with varying topologies - have variable levels of reliability or operational availability.
	EOSD5070#A	ECS shall enable expansion to GByte networks including the ability to provide increased volume of data distribution/access.
	ESN-0815#A	Network simulation and traffic modeling capability shall be provided to troubleshoot network problems and to use in network planning.
	ESN-1060#A	The ESN performance management function shall provide the capability to evaluate the performance of ESN resources and interconnection activities.
	ESN-1065#A	The ESN performance management function shall include trend analysis for prediction of loading and bottlenecks/delays.

Table A-1. Requirements Verification Matrix (93 of 134)

Test Case ID	Requirement ID	Requirement Text
	ESN-1090#A	The ESN shall provide the capability to control the communications performance parameters of the network.
	ESN-1206#A	The ESN capacity and performance shall be consistent with the specified capacity and performance requirements of the ECS functions.
	ESN-1207#A	The ESN capacity and performance shall be capable of expansion to be consistent with the specified capacity and performance growth requirements of the ECS elements and functions.
	ESN-0240#A	The ESN shall be extensible in its design to provide capability for growth and enhancement.
	ESN-0740#A	The ESN network management service shall retrieve performance/fault data about ESN protocol stacks and equipment.
	ESN-0750#A	The ESN shall provide statistical processing capabilities to allow extraction and tabulation of network performance data.
	ESN-0780#A	The network elements including the Internet interfaces, shall have the capability to report, periodically and on an interactive basis, network statistics to the ESN network management function, including the following information: a. Network round trip delay b. Network reset and restart indications c. Outages and CRC errors d. Performance statistics
	EOSD3510#A	Reliability predictions shall be calculated in accordance with the parts count analysis method, Appendix A, of MIL-HDBK-217F, Reliability Prediction of Electronic Equipment.
	EOSD3600#A	Maintainability shall be predicted in accordance with MIL-HDBK-472, Maintainability Prediction, Procedure IV.
	EOSD3625#A	For ECS functions with a backup capability, ECS shall use switch over time to the backup capability in measuring maintainability, rather than down time, when the component goes down.
	SMC-2510#A	The SMC shall provide at a minimum system-wide configuration management for the operational hardware, scientific and system software, and the SMC toolkit contained within ECS. The management system shall support the migration of hardware and software upgrades into the operational environment.
	SMC-6340#A	The SMC shall track system configuration that, at a minimum, audits: a. Hardware resources b. Software resources

Table A-1. Requirements Verification Matrix (94 of 134)

Test Case ID	Requirement ID	Requirement Text
BC009.005	DADS0901#A	The DADS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management h. Distribution and Ingest Management
	PGS-0310#A	The PGS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management c. Accounting Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management.
	EOSD0510#A	ECS shall be capable of being tested during all phases of its development and flight operations.
	EOSD0740#A	Each ECS element shall provide a set of real or simulated functional capabilities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. ECS System (Integration of ECS elements)
	EOSD0750#A	Each ECS element shall provide a set of real or simulated functions which interfaces with both its ECS internal and external entities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. EOSDIS System (Integration of EOSDIS elements)
	EOSD0760#A	Each ECS element shall support end-to-end EOS system testing and fault isolation.
	EOSD0500#A	ECS shall perform the following major functions: a. EOS Mission Planning and Scheduling b. EOS Mission Operations c. Command and Control d. Communications and Networking e. Data Input f. Data Processing g. Data Storage h. Data Distribution i. Information Management j. End-to-End Fault Management k. System Management

Table A-1. Requirements Verification Matrix (95 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD1703#A	ECS shall provide maintenance and operations interfaces to the DAACs to support the functions of: a). System Management b). Science Algorithm Integration c). Product Generation d). Data Archive/Distribution e). User Support Services f). System Maintenance
	SMC-1345#A	The LSM shall perform priority management services to resolve conflicts for ECS resources.
	SMC-3345#A	The LSM shall perform quality assurance for its site/elements performance as well as programmatic areas that includes, at a minimum: a. Quality testing, benchmarks and audits for element enhancement implementations b. Quality checking and audits of products processed and delivered c. Quality testing and audits of element resource performance,
	SMC-3355#A	The LSM shall implement the performance criteria from SMC (including parametric limits and operational threshold levels) for evaluating element resource performance.
	SMC-3375#A	For each limit checked parameter, the LSM (including those thresholds directed by the SMC) shall have the capability of evaluating multiple levels of thresholds including, at a minimum: a. On/off b. Pass/fail c. Various levels of degradation
	SMC-3385#A	The LSM shall evaluate system performance against the ESDIS project established performance criteria.
	SMC-3395#A	The LSM shall generate, in response to each limit check threshold, alert indicators of fault or degraded conditions.
	SMC-3397#A	The LSM shall generate, as needed, requests for performance testing, including, at a minimum: a. Resource to be tested b. Test purpose c. Requested test priority d. Required test environment e. Impacts to operations f. Expected test results
	SMC-3415#A	The LSM shall perform short and long-term trend analysis of element performance, including, at a minimum: a. Operational status b. Performance of a particular resource c. Maintenance activities (e.g., number of repairs per item)

Table A-1. Requirements Verification Matrix (96 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD0545#A	ECS shall be able to accommodate growth (e.g., capacity) in all of its functions as well as the addition of new functions.
	EOSD3490#A	Reliability statistics for ECS shall be collected and monitored using the Mean Time Between Maintenance (MTBM) for each component and operational capability.
	EOSD3492#A	RMA data shall be maintained in a repository accessible for logistics analysis and other purposes.
	EOSD3620#A	ECS shall predict and periodically assess maintainability by measuring the actual MDT and comparing to the required MDT.
	EOSD3630#A	The maximum down time shall not exceed twice the required MDT in 99 percent of failure occurrences.
	EOSD3700#A	ECS functions shall have an operational availability of 0.96 at a minimum (.998 design goal) and an MDT of four (4) hours or less (1.5 hour design goal), unless otherwise specified. The above requirement covers equipment including: a. "Non-critical" equipment configured with the critical equipment supporting the functional capabilities in the requirements. b. Equipment providing other functionality not explicitly stated in the RMA requirements that follow.
	EOSD3710#A	The ECS shall have no single point of failure for functions associated with real-time operations of the spacecraft and instruments.
	EOSD4035#A	The ESN shall have no single point of failure for functions associated with network databases and configuration data.
	EOSD4036#A	The operational availability of individual ESN segments shall be consistent with the specified operational availability of the supported ECS functions.
	EOSD5000#A	ECS shall enable the addition of other data providers, e.g. DAACs, SCFs, ADCs, ODCs, which may: - provide heterogeneous services, i.e. services in support of EOS which may be less than or different than ECS services. - be connected with varying topologies - have variable levels of reliability or operational availability.
	EOSD5070#A	ECS shall enable expansion to GByte networks including the ability to provide increased volume of data distribution/access.
	ESN-0815#A	Network simulation and traffic modeling capability shall be provided to troubleshoot network problems and to use in network planning.
	ESN-1060#A	The ESN performance management function shall provide the capability to evaluate the performance of ESN resources and interconnection activities.
	ESN-1065#A	The ESN performance management function shall include trend analysis for prediction of loading and bottlenecks/delays.

Table A-1. Requirements Verification Matrix (97 of 134)

Test Case ID	Requirement ID	Requirement Text
	ESN-1090#A	The ESN shall provide the capability to control the communications performance parameters of the network.
	ESN-1206#A	The ESN capacity and performance shall be consistent with the specified capacity and performance requirements of the ECS functions.
	ESN-1207#A	The ESN capacity and performance shall be capable of expansion to be consistent with the specified capacity and performance growth requirements of the ECS elements and functions.
	ESN-0240#A	The ESN shall be extensible in its design to provide capability for growth and enhancement.
	ESN-0740#A	The ESN network management service shall retrieve performance/fault data about ESN protocol stacks and equipment.
	ESN-0750#A	The ESN shall provide statistical processing capabilities to allow extraction and tabulation of network performance data.
	ESN-0780#A	The network elements including the Internet interfaces, shall have the capability to report, periodically and on an interactive basis, network statistics to the ESN network management function, including the following information: a. Network round trip delay b. Network reset and restart indications c. Outages and CRC errors d. Performance statistics
	EOSD3510#A	Reliability predictions shall be calculated in accordance with the parts count analysis method, Appendix A, of MIL-HDBK-217F, Reliability Prediction of Electronic Equipment.
	EOSD3600#A	Maintainability shall be predicted in accordance with MIL-HDBK-472, Maintainability Prediction, Procedure IV.
	EOSD3625#A	For ECS functions with a backup capability, ECS shall use switchover time to the backup capability in measuring maintainability, rather than down time, when the component goes down.
	SMC-2305#A	The LSM shall monitor the spares inventory within its element.
	SMC-2315#A	The LSM shall manage the replenishment of spare parts within its element.
	SMC-2325#A	The LSM shall monitor the consumable inventory within its element for items used by the system including, at a minimum: a. Computer tapes b. Computer disks c. Computer paper
	SMC-2335#A	The LSM shall manage the replenishment of consumable items for its element.

Table A-1. Requirements Verification Matrix (98 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-2515#A	The LSM shall provide configuration management for at least the operational hardware, system software, and scientific software within its element and for the migration of enhancements into the operational system.
	SMC-2535#A	Upon approval of an enhancement, the LSM shall facilitate the implementation of the approved changes within an elements hardware and software.
	SMC-6345#A	The LSM shall, as needed, perform configuration accountability to include, at a minimum, the audit of hardware and software resources within its element.
BC009.006	DADS0901#A	The DADS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management h. Distribution and Ingest Management
	PGS-0310#A	The PGS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management c. Accounting Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management.
	EOSD0510#A	ECS shall be capable of being tested during all phases of its development and flight operations.
	EOSD0740#A	Each ECS element shall provide a set of real or simulated functional capabilities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. ECS System (Integration of ECS elements)
	EOSD0750#A	Each ECS element shall provide a set of real or simulated functions which interfaces with both its ECS internal and external entities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. EOSDIS System (Integration of EOSDIS elements)
	EOSD0760#A	Each ECS element shall support end-to-end EOS system testing and fault isolation.

Table A-1. Requirements Verification Matrix (99 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD0500#A	ECS shall perform the following major functions: a. EOS Mission Planning and Scheduling b. EOS Mission Operations c. Command and Control d. Communications and Networking e. Data Input f. Data Processing g. Data Storage h. Data Distribution i. Information Management j. End-to-End Fault Management k. System Management
	EOSD1703#A	ECS shall provide maintenance and operations interfaces to the DAACs to support the functions of: a). System Management b). Science Algorithm Integration c). Product Generation d). Data Archive/Distribution e). User Support Services f). System Maintenance
	SMC-1345#A	The LSM shall perform priority management services to resolve conflicts for ECS resources.
	SMC-3345#A	The LSM shall perform quality assurance for its site/elements performance as well as programmatic areas that includes, at a minimum: a. Quality testing, benchmarks and audits for element enhancement implementations b. Quality checking and audits of products processed and delivered c. Quality testing and audits of element resource performance,
	SMC-3355#A	The LSM shall implement the performance criteria from SMC (including parametric limits and operational threshold levels) for evaluating element resource performance.
	SMC-3375#A	For each limit checked parameter, the LSM (including those thresholds directed by the SMC) shall have the capability of evaluating multiple levels of thresholds including, at a minimum: a. On/off b. Pass/fail c. Various levels of degradation
	SMC-3385#A	The LSM shall evaluate system performance against the ESDIS project established performance criteria.
	SMC-3395#A	The LSM shall generate, in response to each limit check threshold, alert indicators of fault or degraded conditions.

Table A-1. Requirements Verification Matrix (100 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-3397#A	The LSM shall generate, as needed, requests for performance testing, including, at a minimum: a. Resource to be tested b. Test purpose c. Requested test priority d. Required test environment e. Impacts to operations f. Expected test results
	SMC-3415#A	The LSM shall perform short and long-term trend analysis of element performance, including, at a minimum: a. Operational status b. Performance of a particular resource c. Maintenance activities (e.g., number of repairs per item)
	EOSD0545#A	ECS shall be able to accommodate growth (e.g., capacity) in all of its functions as well as the addition of new functions.
	EOSD3490#A	Reliability statistics for ECS shall be collected and monitored using the Mean Time Between Maintenance (MTBM) for each component and operational capability.
	EOSD3492#A	RMA data shall be maintained in a repository accessible for logistics analysis and other purposes.
	EOSD3620#A	ECS shall predict and periodically assess maintainability by measuring the actual MDT and comparing to the required MDT.
	EOSD3630#A	The maximum down time shall not exceed twice the required MDT in 99 percent of failure occurrences.
	EOSD3700#A	ECS functions shall have an operational availability of 0.96 at a minimum (.998 design goal) and an MDT of four (4) hours or less (1.5 hour design goal), unless otherwise specified. The above requirement covers equipment including: a. "Non-critical" equipment configured with the critical equipment supporting the functional capabilities in the requirements. b. Equipment providing other functionality not explicitly stated in the RMA requirements that follow.
	EOSD3710#A	The ECS shall have no single point of failure for functions associated with real-time operations of the spacecraft and instruments.
	EOSD4035#A	The ESN shall have no single point of failure for functions associated with network databases and configuration data.
	EOSD4036#A	The operational availability of individual ESN segments shall be consistent with the specified operational availability of the supported ECS functions.

Table A-1. Requirements Verification Matrix (101 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD5000#A	ECS shall enable the addition of other data providers, e.g. DAACs, SCFs, ADCs, ODCs, which may: - provide heterogeneous services, i.e. services in support of EOS which may be less than or different than ECS services. - be connected with varying topologies - have variable levels of reliability or operational availability.
	EOSD5070#A	ECS shall enable expansion to GByte networks including the ability to provide increased volume of data distribution/access.
	ESN-0815#A	Network simulation and traffic modeling capability shall be provided to troubleshoot network problems and to use in network planning.
	ESN-1060#A	The ESN performance management function shall provide the capability to evaluate the performance of ESN resources and interconnection activities.
	ESN-1065#A	The ESN performance management function shall include trend analysis for prediction of loading and bottlenecks/delays.
	ESN-1090#A	The ESN shall provide the capability to control the communications performance parameters of the network.
	ESN-1206#A	The ESN capacity and performance shall be consistent with the specified capacity and performance requirements of the ECS functions.
	ESN-1207#A	The ESN capacity and performance shall be capable of expansion to be consistent with the specified capacity and performance growth requirements of the ECS elements and functions.
	ESN-0240#A	The ESN shall be extensible in its design to provide capability for growth and enhancement.
	ESN-0740#A	The ESN network management service shall retrieve performance/fault data about ESN protocol stacks and equipment.
	ESN-0750#A	The ESN shall provide statistical processing capabilities to allow extraction and tabulation of network performance data.
	ESN-0780#A	The network elements including the Internet interfaces, shall have the capability to report, periodically and on an interactive basis, network statistics to the ESN network management function, including the following information: a. Network round trip delay b. Network reset and restart indications c. Outages and CRC errors d. Performance statistics
	EOSD3510#A	Reliability predictions shall be calculated in accordance with the parts count analysis method, Appendix A, of MIL-HDBK-217F, Reliability Prediction of Electronic Equipment.
	EOSD3600#A	Maintainability shall be predicted in accordance with MIL-HDBK-472, Maintainability Prediction, Procedure IV.

Table A-1. Requirements Verification Matrix (102 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD3625#A	For ECS functions with a backup capability, ECS shall use switchover time to the backup capability in measuring maintainability, rather than down time, when the component goes down.
BC009.007	DADS0901#A	The DADS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management h. Distribution and Ingest Management
	PGS-0310#A	The PGS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management c. Accounting Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management.
	EOSD0510#A	ECS shall be capable of being tested during all phases of its development and flight operations.
	EOSD0740#A	Each ECS element shall provide a set of real or simulated functional capabilities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. ECS System (Integration of ECS elements)
	EOSD0750#A	Each ECS element shall provide a set of real or simulated functions which interfaces with both its ECS internal and external entities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. EOSDIS System (Integration of EOSDIS elements)
	EOSD0760#A	Each ECS element shall support end-to-end EOS system testing and fault isolation.

Table A-1. Requirements Verification Matrix (103 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD0500#A	ECS shall perform the following major functions: a. EOS Mission Planning and Scheduling b. EOS Mission Operations c. Command and Control d. Communications and Networking e. Data Input f. Data Processing g. Data Storage h. Data Distribution i. Information Management j. End-to-End Fault Management k. System Management
	EOSD1703#A	ECS shall provide maintenance and operations interfaces to the DAACs to support the functions of: a). System Management b). Science Algorithm Integration c). Product Generation d). Data Archive/Distribution e). User Support Services f). System Maintenance
	SMC-1345#A	The LSM shall perform priority management services to resolve conflicts for ECS resources.
	SMC-3345#A	The LSM shall perform quality assurance for its site/elements performance as well as programmatic areas that includes, at a minimum: a. Quality testing, benchmarks and audits for element enhancement implementations b. Quality checking and audits of products processed and delivered c. Quality testing and audits of element resource performance,
	SMC-3355#A	The LSM shall implement the performance criteria from SMC (including parametric limits and operational threshold levels) for evaluating element resource performance.
	SMC-3375#A	For each limit checked parameter, the LSM (including those thresholds directed by the SMC) shall have the capability of evaluating multiple levels of thresholds including, at a minimum: a. On/off b. Pass/fail c. Various levels of degradation
	SMC-3385#A	The LSM shall evaluate system performance against the ESDIS project established performance criteria.
	SMC-3395#A	The LSM shall generate, in response to each limit check threshold, alert indicators of fault or degraded conditions.

Table A-1. Requirements Verification Matrix (104 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-3397#A	The LSM shall generate, as needed, requests for performance testing, including, at a minimum: a. Resource to be tested b. Test purpose c. Requested test priority d. Required test environment e. Impacts to operations f. Expected test results
	SMC-3415#A	The LSM shall perform short and long-term trend analysis of element performance, including, at a minimum: a. Operational status b. Performance of a particular resource c. Maintenance activities (e.g., number of repairs per item)
	EOSD0545#A	ECS shall be able to accommodate growth (e.g., capacity) in all of its functions as well as the addition of new functions.
	EOSD3490#A	Reliability statistics for ECS shall be collected and monitored using the Mean Time Between Maintenance (MTBM) for each component and operational capability.
	EOSD3492#A	RMA data shall be maintained in a repository accessible for logistics analysis and other purposes.
	EOSD3620#A	ECS shall predict and periodically assess maintainability by measuring the actual MDT and comparing to the required MDT.
	EOSD3630#A	The maximum down time shall not exceed twice the required MDT in 99 percent of failure occurrences.
	EOSD3700#A	ECS functions shall have an operational availability of 0.96 at a minimum (.998 design goal) and an MDT of four (4) hours or less (1.5 hour design goal), unless otherwise specified. The above requirement covers equipment including: a. "Non-critical" equipment configured with the critical equipment supporting the functional capabilities in the requirements. b. Equipment providing other functionality not explicitly stated in the RMA requirements that follow.
	EOSD3710#A	The ECS shall have no single point of failure for functions associated with real-time operations of the spacecraft and instruments.
	EOSD4035#A	The ESN shall have no single point of failure for functions associated with network databases and configuration data.
	EOSD4036#A	The operational availability of individual ESN segments shall be consistent with the specified operational availability of the supported ECS functions.

Table A-1. Requirements Verification Matrix (105 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD5000#A	ECS shall enable the addition of other data providers, e.g. DAACs, SCFs, ADCs, ODCs, which may: - provide heterogeneous services, i.e. services in support of EOS which may be less than or different than ECS services. - be connected with varying topologies - have variable levels of reliability or operational availability.
	EOSD5070#A	ECS shall enable expansion to GByte networks including the ability to provide increased volume of data distribution/access.
	ESN-0815#A	Network simulation and traffic modeling capability shall be provided to troubleshoot network problems and to use in network planning.
	ESN-1060#A	The ESN performance management function shall provide the capability to evaluate the performance of ESN resources and interconnection activities.
	ESN-1065#A	The ESN performance management function shall include trend analysis for prediction of loading and bottlenecks/delays.
	ESN-1090#A	The ESN shall provide the capability to control the communications performance parameters of the network.
	ESN-1206#A	The ESN capacity and performance shall be consistent with the specified capacity and performance requirements of the ECS functions.
	ESN-1207#A	The ESN capacity and performance shall be capable of expansion to be consistent with the specified capacity and performance growth requirements of the ECS elements and functions.
	ESN-0240#A	The ESN shall be extensible in its design to provide capability for growth and enhancement.
	ESN-0740#A	The ESN network management service shall retrieve performance/fault data about ESN protocol stacks and equipment.
	ESN-0750#A	The ESN shall provide statistical processing capabilities to allow extraction and tabulation of network performance data.
	ESN-0780#A	The network elements including the Internet interfaces, shall have the capability to report, periodically and on an interactive basis, network statistics to the ESN network management function, including the following information: a. Network round trip delay b. Network reset and restart indications c. Outages and CRC errors d. Performance statistics
	EOSD3510#A	Reliability predictions shall be calculated in accordance with the parts count analysis method, Appendix A, of MIL-HDBK-217F, Reliability Prediction of Electronic Equipment.
	EOSD3600#A	Maintainability shall be predicted in accordance with MIL-HDBK-472, Maintainability Prediction, Procedure IV.

Table A-1. Requirements Verification Matrix (106 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD3625#A	For ECS functions with a backup capability, ECS shall use switchover time to the backup capability in measuring maintainability, rather than down time, when the component goes down.
BC009.009	SMC-6325#A	The LSM shall perform, as needed, data and user audit trails within its element.
	SMC-6335#A	The LSM shall, as needed, maintain and update a data tracking system that, at a minimum: a. Tracks data transport from element input to element output b. Allows the status of all product-production activities to be determined
	ESN-0775#A	The ESN management service shall have the capability to redirect its reports to different devices such as console, disk or printer.
BC009.010	DADS0901#A	The DADS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management h. Distribution and Ingest Management
	PGS-0310#A	The PGS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management c. Accounting Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management.
	EOSD0510#A	ECS shall be capable of being tested during all phases of its development and flight operations.
	EOSD0740#A	Each ECS element shall provide a set of real or simulated functional capabilities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. ECS System (Integration of ECS elements)
	EOSD0750#A	Each ECS element shall provide a set of real or simulated functions which interfaces with both its ECS internal and external entities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. EOSDIS System (Integration of EOSDIS elements)

Table A-1. Requirements Verification Matrix (107 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD0760#A	Each ECS element shall support end-to-end EOS system testing and fault isolation.
	EOSD0500#A	ECS shall perform the following major functions: a. EOS Mission Planning and Scheduling b. EOS Mission Operations c. Command and Control d. Communications and Networking e. Data Input f. Data Processing g. Data Storage h. Data Distribution i. Information Management j. End-to-End Fault Management k. System Management
	EOSD1703#A	ECS shall provide maintenance and operations interfaces to the DAACs to support the functions of: a). System Management b). Science Algorithm Integration c). Product Generation d). Data Archive/Distribution e). User Support Services f). System Maintenance
	NI-0460#A	ECS shall have the capability to receive periodic information regarding NOLAN network performance and link utilization.
	SMC-1345#A	The LSM shall perform priority management services to resolve conflicts for ECS resources.
	SMC-3345#A	The LSM shall perform quality assurance for its site/elements performance as well as programmatic areas that includes, at a minimum: a. Quality testing, benchmarks and audits for element enhancement implementations b. Quality checking and audits of products processed and delivered c. Quality testing and audits of element resource performance,
	SMC-3355#A	The LSM shall implement the performance criteria from SMC (including parametric limits and operational threshold levels) for evaluating element resource performance.
	SMC-3375#A	For each limit checked parameter, the LSM (including those thresholds directed by the SMC) shall have the capability of evaluating multiple levels of thresholds including, at a minimum: a. On/off b. Pass/fail c. Various levels of degradation
	SMC-3385#A	The LSM shall evaluate system performance against the ESDIS project established performance criteria.

Table A-1. Requirements Verification Matrix (108 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-3395#A	The LSM shall generate, in response to each limit check threshold, alert indicators of fault or degraded conditions.
	SMC-3397#A	The LSM shall generate, as needed, requests for performance testing, including, at a minimum: a. Resource to be tested b. Test purpose c. Requested test priority d. Required test environment e. Impacts to operations f. Expected test results
	SMC-3415#A	The LSM shall perform short and long-term trend analysis of element performance, including, at a minimum: a. Operational status b. Performance of a particular resource c. Maintenance activities (e.g., number of repairs per item)
	EOSD0545#A	ECS shall be able to accommodate growth (e.g., capacity) in all of its functions as well as the addition of new functions.
	EOSD3490#A	Reliability statistics for ECS shall be collected and monitored using the Mean Time Between Maintenance (MTBM) for each component and operational capability.
	EOSD3492#A	RMA data shall be maintained in a repository accessible for logistics analysis and other purposes.
	EOSD3620#A	ECS shall predict and periodically assess maintainability by measuring the actual MDT and comparing to the required MDT.
	EOSD3630#A	The maximum down time shall not exceed twice the required MDT in 99 percent of failure occurrences.
	EOSD3700#A	ECS functions shall have an operational availability of 0.96 at a minimum (.998 design goal) and an MDT of four (4) hours or less (1.5 hour design goal), unless otherwise specified. The above requirement covers equipment including: a. "Non-critical" equipment configured with the critical equipment supporting the functional capabilities in the requirements. b. Equipment providing other functionality not explicitly stated in the RMA requirements that follow.
	EOSD3710#A	The ECS shall have no single point of failure for functions associated with real-time operations of the spacecraft and instruments.
	EOSD4035#A	The ESN shall have no single point of failure for functions associated with network databases and configuration data.
	EOSD4036#A	The operational availability of individual ESN segments shall be consistent with the specified operational availability of the supported ECS functions.

Table A-1. Requirements Verification Matrix (109 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD5000#A	ECS shall enable the addition of other data providers, e.g. DAACs, SCFs, ADCs, ODCs, which may: - provide heterogeneous services, i.e. services in support of EOS which may be less than or different than ECS services. - be connected with varying topologies - have variable levels of reliability or operational availability.
	EOSD5070#A	ECS shall enable expansion to GByte networks including the ability to provide increased volume of data distribution/access.
	ESN-0815#A	Network simulation and traffic modeling capability shall be provided to troubleshoot network problems and to use in network planning.
	ESN-1060#A	The ESN performance management function shall provide the capability to evaluate the performance of ESN resources and interconnection activities.
	ESN-1065#A	The ESN performance management function shall include trend analysis for prediction of loading and bottlenecks/delays.
	ESN-1090#A	The ESN shall provide the capability to control the communications performance parameters of the network.
	ESN-1206#A	The ESN capacity and performance shall be consistent with the specified capacity and performance requirements of the ECS functions.
	ESN-1207#A	The ESN capacity and performance shall be capable of expansion to be consistent with the specified capacity and performance growth requirements of the ECS elements and functions.
	ESN-0240#A	The ESN shall be extensible in its design to provide capability for growth and enhancement.
	ESN-0740#A	The ESN network management service shall retrieve performance/fault data about ESN protocol stacks and equipment.
	ESN-0750#A	The ESN shall provide statistical processing capabilities to allow extraction and tabulation of network performance data.
	ESN-0780#A	The network elements including the Internet interfaces, shall have the capability to report, periodically and on an interactive basis, network statistics to the ESN network management function, including the following information: a. Network round trip delay b. Network reset and restart indications c. Outages and CRC errors d. Performance statistics
	EOSD3510#A	Reliability predictions shall be calculated in accordance with the parts count analysis method, Appendix A, of MIL-HDBK-217F, Reliability Prediction of Electronic Equipment.
	EOSD3600#A	Maintainability shall be predicted in accordance with MIL-HDBK-472, Maintainability Prediction, Procedure IV.

Table A-1. Requirements Verification Matrix (110 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD3625#A	For ECS functions with a backup capability, ECS shall use switch-over time to the backup capability in measuring maintainability, rather than down time, when the component goes down.
	EOSD5250#A	ECS shall enable access to configuration controlled applications programming interfaces that permit development of DAAC-unique value added services and products where DAAC-unique value added services may consist of one or more of the following types of developments: a. Visualization utilities and products b. Data sets and inter-data set usability utilities and products c. Data analysis utilities d. Special sub setting capabilities (e.g. dynamic) e. On-line analysis functions f. New search and access techniques g. Data acquisition planning and utilities h. Experimental QA techniques i. Non-digital data utilities and products j. System Management Functions
	ESN-0003#A	The ESN shall enable researchers on existing networks (TCP/IP and GOSIP) to gain access to data and ECS services in a transparent manner to the underlying differences between the networks.
	C-MSS-90020	The DBMS shall support a client-server design paradigm with distributed data allocation.
	SMC-2505#A	The LSM shall update the system-wide inventory data base consisting of all hardware, system software, and scientific software contained within its element.
	C-MSS-90060	The DBMS shall provide an SQL interface with query, update, and administrative functions capabilities.
	C-MSS-90070	The DBMS shall be in compliance with the SQL-2 of Federal Information Processing System Publication (FIPS PUB) 127-1.
	C-MSS-90160	The DBMS shall support features in compliance with X/Open environment to include the following: a. Hardware independence b. Operating systems independence c. Network protocols independence

Table A-1. Requirements Verification Matrix (111 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-90080	The DBMS shall support mathematical operations to generate statistics from management data to include: a._average b._maximum c._minimum d._standard deviation e._sum f._count g._variance
	C-MSS-90290	The DBMS shall provide an audit trail of chronological activities in the database.
	C-MSS-90280	The DBMS shall provide the capability to issue and record a database checkpoint.
	C-MSS-90260	The DBMS shall provide a capability to export, archival, and restore a database.
	C-MSS-90240	The DBMS shall provide for automatic database recovery including a means to: a. automatically restore undamaged portions of a database and recover work in progress after a system or component failure b. achieve dynamic backout of database modifications, performed by a failing transaction, that does not affect separate, concurrent tasks
	C-MSS-90230	The DBMS shall provide a transaction roll backward capability to a specified time or state: a. Restore a database b. Restore all or operator selected database objects of any database
	C-MSS-90210	The DBMS shall support the following features: a. Data compression of nulls and variable length character strings, and indexes b. Space reclaimed from deleted records automatically c. Variable-length column storage
	C-MSS-90200	The DBMS shall perform on-line disk management functions to include: a. Relocation of database files to different disks b. Expansion of database size by adding new physical data files to it on-line c. Dynamic pre-allocation of contiguous space for tables d. Database objects and indexes can span physical files e. Database objects and indexes can exist on different disks
	C-MSS-90190	The DBMS shall provide capabilities for specifying frequency, time, and type of backups.

Table A-1. Requirements Verification Matrix (112 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-90150	The DBMS shall support access structures (i.e., single-level indexes, multilevel indexes) to improve the efficiency of retrieval of management data.
	C-MSS-90140	The DBMS shall support, or be accessed via, CSS session-establishment services.
	SMC-6325#A	The LSM shall perform, as needed, data and user audit trails within its element.
	SMC-6335#A	The LSM shall, as needed, maintain and update a data tracking system that, at a minimum: a. Tracks data transport from element input to element output b. Allows the status of all product-production activities to be determined
	ESN-0775#A	The ESN management service shall have the capability to redirect its reports to different devices such as console, disk or printer.
BC009.011	DADS0901#A	The DADS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management h. Distribution and Ingest Management
	PGS-0310#A	The PGS element shall collect the management data used to support the following system management functions: a. Fault Management b. Configuration Management c. Accounting Management d. Accountability Management e. Performance Management f. Security Management g. Scheduling Management.
	EOSD0510#A	ECS shall be capable of being tested during all phases of its development and flight operations.
	EOSD0740#A	Each ECS element shall provide a set of real or simulated functional capabilities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. ECS System (Integration of ECS elements)

Table A-1. Requirements Verification Matrix (113 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD0750#A	Each ECS element shall provide a set of real or simulated functions which interfaces with both its ECS internal and external entities for use in the following types of test: a. Subsystem (components of an ECS element) b. Element (fully integrated element) c. EOSDIS System (Integration of EOSDIS elements)
	EOSD0760#A	Each ECS element shall support end-to-end EOS system testing and fault isolation.
	EOSD0500#A	ECS shall perform the following major functions: a. EOS Mission Planning and Scheduling b. EOS Mission Operations c. Command and Control d. Communications and Networking e. Data Input f. Data Processing g. Data Storage h. Data Distribution i. Information Management j. End-to-End Fault Management k. System Management
	EOSD1703#A	ECS shall provide maintenance and operations interfaces to the DAACs to support the functions of: a). System Management b). Science Algorithm Integration c). Product Generation d). Data Archive/Distribution e). User Support Services f). System Maintenance
	NI-0460#A	ECS shall have the capability to receive periodic information regarding NOLAN network performance and link utilization.
	SMC-1345#A	The LSM shall perform priority management services to resolve conflicts for ECS resources.
	SMC-3345#A	The LSM shall perform quality assurance for its site/elements performance as well as programmatic areas that includes, at a minimum: a. Quality testing, benchmarks and audits for element enhancement implementations b. Quality checking and audits of products processed and delivered c. Quality testing and audits of element resource performance,
	SMC-3355#A	The LSM shall implement the performance criteria from SMC (including parametric limits and operational threshold levels) for evaluating element resource performance.

Table A-1. Requirements Verification Matrix (114 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-3375#A	For each limit checked parameter, the LSM (including those thresholds directed by the SMC) shall have the capability of evaluating multiple levels of thresholds including, at a minimum: a. On/off b. Pass/fail c. Various levels of degradation
	SMC-3385#A	The LSM shall evaluate system performance against the ESDIS project established performance criteria.
	SMC-3395#A	The LSM shall generate, in response to each limit check threshold, alert indicators of fault or degraded conditions.
	SMC-3397#A	The LSM shall generate, as needed, requests for performance testing, including, at a minimum: a. Resource to be tested b. Test purpose c. Requested test priority d. Required test environment e. Impacts to operations f. Expected test results
	SMC-3415#A	The LSM shall perform short and long-term trend analysis of element performance, including, at a minimum: a. Operational status b. Performance of a particular resource c. Maintenance activities (e.g., number of repairs per item)
	EOSD0545#A	ECS shall be able to accommodate growth (e.g., capacity) in all of its functions as well as the addition of new functions.
	EOSD3490#A	Reliability statistics for ECS shall be collected and monitored using the Mean Time Between Maintenance (MTBM) for each component and operational capability.
	EOSD3492#A	RMA data shall be maintained in a repository accessible for logistics analysis and other purposes.
	EOSD3620#A	ECS shall predict and periodically assess maintainability by measuring the actual MDT and comparing to the required MDT.
	EOSD3630#A	The maximum down time shall not exceed twice the required MDT in 99 percent of failure occurrences.
	EOSD3700#A	ECS functions shall have an operational availability of 0.96 at a minimum (.998 design goal) and an MDT of four (4) hours or less (1.5 hour design goal), unless otherwise specified. The above requirement covers equipment including: a. "Non-critical" equipment configured with the critical equipment supporting the functional capabilities in the requirements. b. Equipment providing other functionality not explicitly stated in the RMA requirements that follow.

Table A-1. Requirements Verification Matrix (115 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD3710#A	The ECS shall have no single point of failure for functions associated with real-time operations of the spacecraft and instruments.
	EOSD4035#A	The ESN shall have no single point of failure for functions associated with network databases and configuration data.
	EOSD4036#A	The operational availability of individual ESN segments shall be consistent with the specified operational availability of the supported ECS functions.
	EOSD5000#A	ECS shall enable the addition of other data providers, e.g. DAACs, SCFs, ADCs, ODCs, which may: - provide heterogeneous services, i.e. services in support of EOS which may be less than or different than ECS services. - be connected with varying topologies - have variable levels of reliability or operational availability.
	EOSD5070#A	ECS shall enable expansion to GByte networks including the ability to provide increased volume of data distribution/access.
	ESN-0815#A	Network simulation and traffic modeling capability shall be provided to troubleshoot network problems and to use in network planning.
	ESN-1060#A	The ESN performance management function shall provide the capability to evaluate the performance of ESN resources and interconnection activities.
	ESN-1065#A	The ESN performance management function shall include trend analysis for prediction of loading and bottlenecks/delays.
	ESN-1090#A	The ESN shall provide the capability to control the communications performance parameters of the network.
	ESN-1206#A	The ESN capacity and performance shall be consistent with the specified capacity and performance requirements of the ECS functions.
	ESN-1207#A	The ESN capacity and performance shall be capable of expansion to be consistent with the specified capacity and performance growth requirements of the ECS elements and functions.
	ESN-0240#A	The ESN shall be extensible in its design to provide capability for growth and enhancement.
	ESN-0740#A	The ESN network management service shall retrieve performance/fault data about ESN protocol stacks and equipment.
	ESN-0750#A	The ESN shall provide statistical processing capabilities to allow extraction and tabulation of network performance data.

Table A-1. Requirements Verification Matrix (116 of 134)

Test Case ID	Requirement ID	Requirement Text
	ESN-0780#A	The network elements including the Internet interfaces, shall have the capability to report, periodically and on an interactive basis, network statistics to the ESN network management function, including the following information: a. Network round trip delay b. Network reset and restart indications c. Outages and CRC errors d. Performance statistics
	EOSD3510#A	Reliability predictions shall be calculated in accordance with the parts count analysis method, Appendix A, of MIL-HDBK-217F, Reliability Prediction of Electronic Equipment.
	EOSD3600#A	Maintainability shall be predicted in accordance with MIL-HDBK-472, Maintainability Prediction, Procedure IV.
	EOSD3625#A	For ECS functions with a backup capability, ECS shall use switch-over time to the backup capability in measuring maintainability, rather than down time, when the component goes down.
	SMC-8305#A	The LSM shall have the same report generator capability as for the SMC, except it shall be limited to generating reports covering only its particular site or its particular element.
	SMC-2115#A	The LSM shall convey for site or element implementation, the managerial and operational directives regarding the allocation or upgrade of any elements hardware and scientific and systems software.
	SMC-2415#A	The LSM shall receive from the SMC descriptions and schedules for training courses.
	C-MSS-90520	The Report Generator shall have the capability to generate ad hoc reports from management data maintained in the DBMS.
	C-MSS-90530	The Report Generator shall provide the capability to format reports to include the report: a._title b._header c._footer d._page number e._date/time of report
	C-MSS-90500	The Report Generator shall be compatible with the DBMS.
	C-MSS-90570	The Report Generator shall have the capability to generate charts and graphs (e.g., bar, pie, line, etc.) from management data maintained in the DBMS.
	C-MSS-90600	The Report Generator shall provide the capability to redirect generated reports to: a._console b._disk file c._printer

Table A-1. Requirements Verification Matrix (117 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-MSS-91010	The MSS Office Automation word processing capability shall facilitate the: a._preparation, revision, and recording of documents, messages, reports, and data b._import, transformation, and editing of documents produced by other word processing packages c._insertion of worksheet and graphic images into documents, messages, and reports d._transfer of document, message, and report information to spreadsheet and graphics applications e._printing of documents, messages, reports, and data
	C-MSS-90510	The Report Generator shall provide a Motif based Graphical User Interface (GUI) for creating ad hoc reports.
TC032.001	C-CSS-61010	The CSS Electronic Mail Service shall interoperate and exchange messages with external mail systems based on SMTP and X.400 protocols.
	C-CSS-61020	The CSS Electronic Mail Service shall be capable of sending and receiving the Multi-purpose Internet Mail Extensions (MIME) messages.
	C-CSS-61030	The CSS Electronic Mail Service shall use the existing X.400 gateway available at GSFC to support X.400 operations.
	C-CSS-61290	The CSS Electronic Mail Service shall provide functionality to send reply for a received message to a. the author b. to all destinations addressed in the incoming message MailTool
TC032.002	C-CSS-61310	The CSS Electronic Mail Service shall provide a MAILBOX where all incoming messages for operators will be stored.
	C-CSS-61320	The CSS Electronic Mail Service shall provide operator defined folders to store messages for long term archive.
	C-CSS-61330	The CSS Electronic Mail Service shall allow copying and/or moving messages from the MAILBOX to the operator specified folders.
	C-CSS-61360	The CSS Electronic Mail Service shall be capable of showing a summary of all messages in the MAILBOX or in a folder which minimally contains: a. title/subject of the message b. name of the author c. date/time of the message origination
	C-CSS-61370	The CSS Electronic Mail Service shall provide an editor to compose a message.
	C-CSS-61380	The CSS Electronic Mail Service shall provide a title/subject field for a message.

Table A-1. Requirements Verification Matrix (118 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-CSS-61420	The CSS Electronic Mail Service shall provide a capability to maintain private mailing lists (each list may contain multiple destination) for individual operators.
	C-CSS-61430	The CSS Electronic Mail Service shall allow attaching either text or binary files to a message.
	C-CSS-61440	The CSS Electronic Mail Service shall allow discarding message(s) from the MAILBOX without saving.
	C-CSS-61450	The CSS Electronic Mail Service shall have the capability to forward a message.
	C-CSS-61460	The CSS Electronic Mail Service shall allow cut/copy/paste/delete/undo operations in the editor.
	C-CSS-61470	The CSS Electronic Mail Service shall provide navigation methods to go the next or previous message in the MAILBOX or selected folder.
	C-CSS-61490	The CSS Electronic Mail Service shall provide the capability to search for keywords in messages.
	C-CSS-61500	The CSS Electronic Mail Service shall provide the capability to search the MAILBOX or a folder for keywords in title text.
	C-CSS-61510	The CSS Electronic Mail Service shall provide the capability to search the MAILBOX or folders for a specific author.
TC032.003	C-CSS-61390	The CSS Electronic Mail Service shall allow a message to be sent to multiple destinations.
	C-CSS-61400	The CSS Electronic Mail Service shall allow destinations of the following types: a. a single user b. a position which may be managed by one or many operators c. a site which may consists of several operators.
	C-CSS-61410	The CSS Electronic Mail Service shall provide a capability to maintain public mailing lists (each list may contain multiple destination) which are accessible to all operators.
	C-CSS-61520	The CSS Electronic Mail Service shall accept mailing lists as valid destinations.
	C-CSS-61800	The CSS Electronic Mail Service shall provide the capability to send an electronic mail message non-interactively from an application.
	C-CSS-61810	The CSS Electronic Mail Service shall allow attaching multiple text or binary files to the mail message.
	C-CSS-61820	The CSS Electronic Mail Service shall accept a file name as input for the message text.
	C-CSS-61840	The CSS Electronic Mail Service shall be capable of sending a message to multiple destinations.

Table A-1. Requirements Verification Matrix (119 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-CSS-61850	The CSS Electronic Mail Service shall accept mailing lists as valid destinations.
TC032.004	C-CSS-62050	The CSS Bulletin Board Service shall host the user registration service.
	C-CSS-62070	The CSS Bulletin Board Service shall support download of ECS toolkits.
	C-CSS-62080	The CSS Bulletin Board Service shall collect and maintain access history and statistical information for the service.
	C-CSS-62130	The CSS Bulletin Board Service shall provide a "What's new" feature which informs the user of the new information available on the bulletin boards.
TC032.005	C-CSS-62130	The CSS Bulletin Board Service shall provide a "What's new" feature which informs the user of the new information available on the bulletin boards.
TC032.006	C-CSS-62810	The CSS Bulletin Board Service shall allow attaching ASCII and binary files to a message.
	C-CSS-62820	The CSS Bulletin Board Service shall allow a message to be posted to multiple bulletin boards.
TC033.001	C-CSS-63050	The CSS Virtual Terminal shall support kerberized version of the telnet protocol for secure authentication of users
TC033.002	C-CSS-63060	The CSS Virtual Terminal shall support X-applications
TC034.001	C-CSS-60900	The CSS File Access Service shall provide an API which allows applications to transfer files.
	C-CSS-60910	The CSS File Access Service shall allow for file type selection (ASCII or Binary).
	C-CSS-60920	The CSS File Access Service shall accept authentication information for file transfers.
TC034.002	C-CSS-60810	The CSS File Access Service shall log result of the non-interactive operations to operator specified log files.
	C-CSS-60820	The CSS File Access Service shall provide an option to send alarms and generate events if a scheduled operation fails.
TC034.003	C-CSS-60900	The CSS File Access Service shall provide an API which allows applications to transfer files.
	C-CSS-60920	The CSS File Access Service shall accept authentication information for file transfers.
TC034.004	C-CSS-60810	The CSS File Access Service shall log result of the non-interactive operations to operator specified log files.
	C-CSS-60820	The CSS File Access Service shall provide an option to send alarms and generate events if a scheduled operation fails.
	C-CSS-60530	The CSS File Access Service shall support the kerberized version of File Transfer Protocol for secured file transfers.

Table A-1. Requirements Verification Matrix (120 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-CSS-60800	The CSS File Access Service shall provide an option for scheduling file transfers in a batch mode.
BC010.002	ESN-0010#A	ESN shall provide the following standard services: a. Data Transfer and Management Services b. Electronic Messaging Service c. Remote Terminal Service d. Process to Process Communication Service e. Directory and User Access Control Service f. Network Management Service g. Network Security and Access Control Service h. Internetwork Interface Services i. Bulletin Board Service
	ESN-0340#A	The ESN shall interoperate and exchange messages and data with external SMTP and X.400 mail systems.
	ESN-0350#A	The Electronic Messaging Service, shall be capable of exchanging binary data.
	EOSD0500#A	ECS shall perform the following major functions: a. EOS Mission Planning and Scheduling b. EOS Mission Operations c. Command and Control d. Communications and Networking e. Data Input f. Data Processing g. Data Storage h. Data Distribution i. Information Management j. End-to-End Fault Management k. System Management
BC010.003	ESN-0010#A	ESN shall provide the following standard services: a. Data Transfer and Management Services b. Electronic Messaging Service c. Remote Terminal Service d. Process to Process Communication Service e. Directory and User Access Control Service f. Network Management Service g. Network Security and Access Control Service h. Internetwork Interface Services i. Bulletin Board Service

Table A-1. Requirements Verification Matrix (121 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD0500#A	ECS shall perform the following major functions: a. EOS Mission Planning and Scheduling b. EOS Mission Operations c. Command and Control d. Communications and Networking e. Data Input f. Data Processing g. Data Storage h. Data Distribution i. Information Management j. End-to-End Fault Management k. System Management
BC010.004	AM1-0220#A	The ECS shall have the capability to provide and the MISR, MOPITT, MODIS, and CERES PIs/TLs shall have the capability to receive IST toolkit software, IST toolkit software upgrades, and IST toolkit documentation.
	ESN-0010#A	ESN shall provide the following standard services: a. Data Transfer and Management Services b. Electronic Messaging Service c. Remote Terminal Service d. Process to Process Communication Service e. Directory and User Access Control Service f. Network Management Service g. Network Security and Access Control Service h. Internetwork Interface Services i. Bulletin Board Service
	ESN-1181#A	The ESN shall provide an ECS Bulletin Board capability.
	EOSD0500#A	ECS shall perform the following major functions: a. EOS Mission Planning and Scheduling b. EOS Mission Operations c. Command and Control d. Communications and Networking e. Data Input f. Data Processing g. Data Storage h. Data Distribution i. Information Management j. End-to-End Fault Management k. System Management

Table A-1. Requirements Verification Matrix (122 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-ISS-01020	The ISS shall interface with NSI or an alternate Internet provider at GSFC, LaRC, and EDC to provide DAAC access to science users in accordance with the following documents: a. DID 220, "Communications Requirements for the ECS Project. " b. Interface Requirements Document betweeen EOSDIS Core System (ECS) and the NASA Science Internet (NSI), 194-219-SE1-001
	C-ISS-01100	The ISS shall provide for connectivity with TSDIS in order to transfer TRMM data to the GSFC DAAC.
	C-ISS-01130	The ISS shall provide for connectivity to the LaRC campus network to enabled transfer of data between SCF(s) located at LaRC and the LaRC DAAC.
	C-ISS-01140	The ISS shall connect to the GSFC campus network to enable transfer of data between SCF(s) located at GSFC and the GSFC DAAC.
	C-ISS-01150	The ISS shall provide connectivity between the Landsat system and the EDC DAAC to support the ingest of Landsat data.
	C-ISS-01170	The ISS shall provide connectivity between the EOC and EBNet for AM-1 interface testing.
	C-ISS-01180	The ISS shall provide connectivity between the EOC and EBNet for AM-1 interface testing of EOC/IST communications.
	C-ISS-01185	The ISS shall provide for connectivity to designated international partner (IP) pickup point for ASTER.
	C-ISS-01190	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between EOC components (in support of FOS interface testing at Release A.
	C-ISS-01210	The ISS shall provide a separate network to support functions that will not interfere with the EOCs operational LAN.
	C-ISS-01215	The EOC 's support LAN architecture shall be identical in function and performance to that of the operational network.
	C-ISS-01220	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between SDPS components at the GSFC DAAC.
	C-ISS-01230	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between SDPS components at the LaRC DAAC.
	C-ISS-01240	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between SDPS components at the EDC DAAC.
	C-ISS-01255	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between CSMS components at the GSFC DAAC.
	C-ISS-01260	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between CSMS components at the SMC.
	C-ISS-01270	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between the SMC and the GSFC DAAC.

Table A-1. Requirements Verification Matrix (123 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-ISS-01280	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between the SMC and the EOC.
	C-ISS-01290	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between the FOS EOC components and the CSMS-provided LSM within the EOC.
	C-ISS-01330	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between CSMS components at the LaRC DAAC.
	C-ISS-01340	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between CSMS and SDPS components at the LaRC DAAC.
BC010.005	ESN-0010#A	ESN shall provide the following standard services: a. Data Transfer and Management Services b. Electronic Messaging Service c. Remote Terminal Service d. Process to Process Communication Service e. Directory and User Access Control Service f. Network Management Service g. Network Security and Access Control Service h. Internetwork Interface Services i. Bulletin Board Service
	EOSD0500#A	ECS shall perform the following major functions: a. EOS Mission Planning and Scheduling b. EOS Mission Operations c. Command and Control d. Communications and Networking e. Data Input f. Data Processing g. Data Storage h. Data Distribution i. Information Management j. End-to-End Fault Management k. System Management
	C-ISS-01020	The ISS shall interface with NSI or an alternate Internet provider at GSFC, LaRC, and EDC to provide DAAC access to science users in accordance with the following documents: a. DID 220, "Communications Requirements for the ECS Project. " b. Interface Requirements Document betweeen EOSDIS Core System (ECS) and the NASA Science Internet (NSI), 194-219-SE1-001
	C-ISS-01100	The ISS shall provide for connectivity with TSDIS in order to transfer TRMM data to the GSFC DAAC.
	C-ISS-01130	The ISS shall provide for connectivity to the LaRC campus network to enabled transfer of data between SCF(s) located at LaRC and the LaRC DAAC.

Table A-1. Requirements Verification Matrix (124 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-ISS-01140	The ISS shall connect to the GSFC campus network to enable transfer of data between SCF(s) located at GSFC and the GSFC DAAC.
	C-ISS-01150	The ISS shall provide connectivity between the Landsat system and the EDC DAAC to support the ingest of Landsat data.
	C-ISS-01170	The ISS shall provide connectivity between the EOC and EBNet for AM-1 interface testing.
	C-ISS-01180	The ISS shall provide connectivity between the EOC and EBNet for AM-1 interface testing of EOC/IST communications.
	C-ISS-01185	The ISS shall provide for connectivity to designated international partner (IP) pickup point for ASTER.
	C-ISS-01190	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between EOC components (in support of FOS interface testing at Release A.
	C-ISS-01210	The ISS shall provide a separate network to support functions that will not interfere with the EOCs operational LAN.
	C-ISS-01215	The EOC 's support LAN architecture shall be identical in function and performance to that of the operational network.
	C-ISS-01220	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between SDPS components at the GSFC DAAC.
	C-ISS-01230	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between SDPS components at the LaRC DAAC.
	C-ISS-01240	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between SDPS components at the EDC DAAC.
	C-ISS-01255	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between CSMS components at the GSFC DAAC.
	C-ISS-01260	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between CSMS components at the SMC.
	C-ISS-01270	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between the SMC and the GSFC DAAC.
	C-ISS-01280	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between the SMC and the EOC.
	C-ISS-01290	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between the FOS EOC components and the CSMS-provided LSM within the EOC.
	C-ISS-01330	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between CSMS components at the LaRC DAAC.
	C-ISS-01340	The ISS shall provide LAN connectivity and OSI Layer 1 through Layer 4 services between CSMS and SDPS components at the LaRC DAAC.

Table A-1. Requirements Verification Matrix (125 of 134)

Test Case ID	Requirement ID	Requirement Text
BC010.006	ESN-0010#A	ESN shall provide the following standard services: a. Data Transfer and Management Services b. Electronic Messaging Service c. Remote Terminal Service d. Process to Process Communication Service e. Directory and User Access Control Service f. Network Management Service g. Network Security and Access Control Service h. Internetwork Interface Services i. Bulletin Board Service
	EOSD0500#A	ECS shall perform the following major functions: a. EOS Mission Planning and Scheduling b. EOS Mission Operations c. Command and Control d. Communications and Networking e. Data Input f. Data Processing g. Data Storage h. Data Distribution i. Information Management j. End-to-End Fault Management k. System Management
BC010.007	ESN-0010#A	ESN shall provide the following standard services: a. Data Transfer and Management Services b. Electronic Messaging Service c. Remote Terminal Service d. Process to Process Communication Service e. Directory and User Access Control Service f. Network Management Service g. Network Security and Access Control Service h. Internetwork Interface Services i. Bulletin Board Service
BC010.008	ESN-0010#A	ESN shall provide the following standard services: a. Data Transfer and Management Services b. Electronic Messaging Service c. Remote Terminal Service d. Process to Process Communication Service e. Directory and User Access Control Service f. Network Management Service g. Network Security and Access Control Service h. Internetwork Interface Services i. Bulletin Board Service
	ESN-0290#A	The file transfer and management service shall be available in interactive and non-interactive services.

Table A-1. Requirements Verification Matrix (126 of 134)

Test Case ID	Requirement ID	Requirement Text
	ESN-0300#A	The file transfer and management non-interactive services shall be able to be scheduled.
BC010.009	ESN-0010#A	ESN shall provide the following standard services: a. Data Transfer and Management Services b. Electronic Messaging Service c. Remote Terminal Service d. Process to Process Communication Service e. Directory and User Access Control Service f. Network Management Service g. Network Security and Access Control Service h. Internetwork Interface Services i. Bulletin Board Service
	ESN-0290#A	The file transfer and management service shall be available in interactive and non-interactive services.
	ESN-0590#A	The ESN Directory Service shall be protected by access control capabilities.
	ESN-1380#A	The ESN shall provide countermeasures for the following security threats related to data communications: a. modification of data (i.e., manipulation) while in transit over the network b. disclosure of authentication information c. degradation in network or processing resource performance through denial of service attack d. Impersonation of authentication credentials or authorization privileges.
BC010.010	ESN-0010#A	ESN shall provide the following standard services: a. Data Transfer and Management Services b. Electronic Messaging Service c. Remote Terminal Service d. Process to Process Communication Service e. Directory and User Access Control Service f. Network Management Service g. Network Security and Access Control Service h. Internetwork Interface Services i. Bulletin Board Service
	ESN-0290#A	The file transfer and management service shall be available in interactive and non-interactive services.
BC010.011	AM1-0220#A	The ECS shall have the capability to provide and the MISR, MOPITT, MODIS, and CERES PIs/TLs shall have the capability to receive IST toolkit software, IST toolkit software upgrades, and IST toolkit documentation.

Table A-1. Requirements Verification Matrix (127 of 134)

Test Case ID	Requirement ID	Requirement Text
	ESN-0010#A	ESN shall provide the following standard services: a. Data Transfer and Management Services b. Electronic Messaging Service c. Remote Terminal Service d. Process to Process Communication Service e. Directory and User Access Control Service f. Network Management Service g. Network Security and Access Control Service h. Internetwork Interface Services i. Bulletin Board Service
	ESN-0490#A	The ESN shall provide a name-to-attribute mapping Directory Service at a minimum.
	ESN-0510#A	The directory service shall be able to respond to request for information concerning named objects, either physical or logical, so as to support communication with those objects.
	ESN-0600#A	The ESN Directory service shall include services and supporting mechanisms to authenticate the credentials of a user for the purpose of granting access rights and authorizing requested operations.
	ESN-0610#A	The ESN shall include multiple Directory Service Agents (DSAs) which shall be collectively responsible for holding or retrieving all directory information which is needed by ECS.
	ESN-1181#A	The ESN shall provide an ECS Bulletin Board capability.
	ESN-1380#A	The ESN shall provide countermeasures for the following security threats related to data communications: a. modification of data (i.e., manipulation) while in transit over the network b. disclosure of authentication information c. degradation in network or processing resource performance through denial of service attack d. Impersonation of authentication credentials or authorization privileges.
	ESN-1400#A	The following security functions and services, at a minimum, shall be provided: a. authentication b. access (authorization) control c. data integrity d. data confidentiality

Table A-1. Requirements Verification Matrix (128 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD2510#A	ECS elements shall maintain an audit trail of: a. All accesses to the element security controlled data b. Users/processes/elements requesting access to element security controlled data c. Data access/manipulation operations performed on security controlled data d. Date and time of access to security controlled data e. Unsuccessful access attempt to the element security controlled data by unauthorized users/elements/processes f. Detected computer system viruses and worms g. Actions taken to contain or destroy a virus
	EOSD2620#A	ECS elements shall disconnect a user/element after a predetermined number of unsuccessful attempts to access data.
	EOSD2640#A	ECS elements shall relinquish a connection between the element and a user when the user has not been active for a configurable period of time.
BC010.012	ESN-0010#A	ESN shall provide the following standard services: a. Data Transfer and Management Services b. Electronic Messaging Service c. Remote Terminal Service d. Process to Process Communication Service e. Directory and User Access Control Service f. Network Management Service g. Network Security and Access Control Service h. Internetwork Interface Services i. Bulletin Board Service
	ESN-0290#A	The file transfer and management service shall be available in interactive and non-interactive services.
	ESN-0300#A	The file transfer and management non-interactive services shall be able to be scheduled.
	ESN-0340#A	The ESN shall interoperate and exchange messages and data with external SMTP and X.400 mail systems.
	ESN-0345#A	The ESN shall be capable of transparently transmitting Multi-purpose Internet Mail Extensions (MIME) messages.
	ESN-0350#A	The Electronic Messaging Service, shall be capable of exchanging binary data.
	ESN-0450#A	The ESN shall provide process-to-process communication service.
	ESN-0590#A	The ESN Directory Service shall be protected by access control capabilities.
	EOSD2480#A	ECS elements shall require unique sessions when security controlled data are being manipulated.

Table A-1. Requirements Verification Matrix (129 of 134)

Test Case ID	Requirement ID	Requirement Text
	EOSD5010#A	ECS shall enable extended provider support, i.e. client access of data and services at SCFs and DAACs, as authorized, without distinction to the client.
BC010.013	ESN-0010#A	ESN shall provide the following standard services: a. Data Transfer and Management Services b. Electronic Messaging Service c. Remote Terminal Service d. Process to Process Communication Service e. Directory and User Access Control Service f. Network Management Service g. Network Security and Access Control Service h. Internetwork Interface Services i. Bulletin Board Service
	ESN-0290#A	The file transfer and management service shall be available in interactive and non-interactive services.
	ESN-0300#A	The file transfer and management non-interactive services shall be able to be scheduled.
	ESN-0340#A	The ESN shall interoperate and exchange messages and data with external SMTP and X.400 mail systems.
	ESN-0345#A	The ESN shall be capable of transparently transmitting Multi-purpose Internet Mail Extensions (MIME) messages.
	ESN-0350#A	The Electronic Messaging Service, shall be capable of exchanging binary data.
	ESN-0450#A	The ESN shall provide process-to-process communication service.
	ESN-0590#A	The ESN Directory Service shall be protected by access control capabilities.
	EOSD2480#A	ECS elements shall require unique sessions when security controlled data are being manipulated.
	EOSD5010#A	ECS shall enable extended provider support, i.e. client access of data and services at SCFs and DAACs, as authorized, without distinction to the client.
TC039.004	C-MSS-40460	The MSS configuration management application service at the SMC shall assemble unlicensed toolkit software files for posting to the ECS bulletin board. Files consist of: a. source code; b. linkable object code for selected workstation configurations; c. makefiles that automate installation; d. installation instructions.
	AM1-0220	The ECS shall have the capability to provide and the MISR, MOPITT, MODIS, and CERES PIs/TLs shall have the capability to receive IST toolkit software, IST toolkit software upgrades, and IST toolkit documentation.

Table A-1. Requirements Verification Matrix (130 of 134)

Test Case ID	Requirement ID	Requirement Text
TC040.001	C-CSS-27010	The Process Framework shall accept basic process information that is needed while starting up client and server applications using a set of variables defined in a configuration file.
	C-CSS-27050	The Process Framework shall provide the ability for a process to get the following information: Mode of operation Delta Time Executable Name Process ID Application ID Program ID Major Version Minor Version
TC040.002	C-CSS-27010	The Process Framework shall accept basic process information that is needed while starting up client and server applications using a set of variables defined in a configuration file.
	C-CSS-27020	The Process Framework shall also accept the above variables from the command line.
	C-CSS-27030	The Process Framework shall always provide a higher precedence to the variables defined on the command line.
	C-CSS-27040	The Process Framework shall exit with an error status if the mode of operation and the configuration file name are not provided on the command line.
	C-CSS-27050	The Process Framework shall provide the ability for a process to get the following information: a. Mode of operation b. Delta Time c. Executable Name d. Process ID e. Application ID f. Program ID g. Major Version h. Minor Version
TC040.003	C-CSS-27010	The Process Framework shall accept basic process information that is needed while starting up client and server applications using a set of variables defined in a configuration file.
	C-CSS-27060	The Process Framework shall provide interfaces to the underlying distributed architecture to set the following naming parameters: Short name for server Profile name Group name
	C-CSS-27070	The Process Framework shall provide interfaces to the underlying distributed architecture to establish the server identity.

Table A-1. Requirements Verification Matrix (131 of 134)

Test Case ID	Requirement ID	Requirement Text
	C-CSS-27080	The Process Framework shall provide interfaces to the underlying distributed infrastructure to set the following security parameters: Server principal name Keytab file name ACL database filename or sybase key
	C-CSS-27090	The Process Framework shall provide an interface to the underlying distributed architecture to set the protocol policy.
	C-CSS-27100	The Process Framework shall provide an interface to the underlying distributed architecture to set the host policy.
TC040.004	C-CSS-27010	The Process Framework shall accept basic process information that is needed while starting up client and server applications using a set of variables defined in a configuration file.
	C-CSS-27060	The Process Framework shall provide interfaces to the underlying distributed architecture to set the following naming parameters: Short name for server Profile name Group name
	C-CSS-27070	The Process Framework shall provide interfaces to the underlying distributed architecture to establish the server identity.
	C-CSS-27080	The Process Framework shall provide interfaces to the underlying distributed infrastructure to set the following security parameters: Server principal name Keytab file name ACL database filename or sybase key
	C-CSS-27090	The Process Framework shall provide an interface to the underlying distributed architecture to set the protocol policy.
	C-CSS-27100	The Process Framework shall provide an interface to the underlying distributed architecture to set the host policy.
	C-CSS-27120	The Process Framework shall provide interfaces to the FTP batch processing facility.
	C-CSS-27130	The Process Framework shall provide interfaces to the Server Request Framework.
	C-CSS-27140	The Process Framework shall provide interfaces to the Asynchronous Message Passing service.
	C-CSS-27150	The Process Framework shall interface Management Agent framework to shutdown an application.
TC040.005	C-CSS-27110	The Process Framework shall provide the ability to log errors and events for both client and server processes.
BC011.001	SMC-2320#A	The SMC shall monitor the consumable inventory within each element for items used by the system including, at a minimum: a. Computer tapes b. Computer disks c. Computer paper

Table A-1. Requirements Verification Matrix (132 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-2500#A	The SMC shall establish and maintain a system-wide inventory of all hardware, scientific and system software contained within ECS, including at a minimum: a. Hardware or software identification numbers b. Version numbers and dates c. Manufacturer d. Part number e. Serial number f. Name and locator information for software maintenance g. Location where hardware or software is used
	SMC-2505#A	The LSM shall update the system-wide inventory data base consisting of all hardware, system software, and scientific software contained within its element.
	SMC-2510#A	The SMC shall provide at a minimum system-wide configuration management for the operational hardware, scientific and system software, and the SMC toolkit contained within ECS. The management system shall support the migration of hardware and software upgrades into the operational environment.
	SMC-2515#A	The LSM shall provide configuration management for at least the operational hardware, system software, and scientific software within its element and for the migration of enhancements into the operational system.
	SMC-2520#A	The SMC shall evaluate received system enhancement requests to determine, at a minimum: a. Technical feasibility b. Implementation schedule c. Expected costs d. Existing system-wide hardware and software impacts
	SMC-2530#A	Upon approval of a system enhancement, the SMC shall provide overall management of the implementation of the approved changes to the hardware and system software.
	SMC-2535#A	Upon approval of an enhancement, the LSM shall facilitate the implementation of the approved changes within an elements hardware and software.
	SMC-2540#A	Upon approval to include a fully tested enhancement to the algorithms, the SMC shall provide overall management of the implementation of the approved and modified software into the operational environment.
	ESN-0690#A	The ESN shall be capable of reconfiguration transparent to network users.
BC011.002	SMC-0340#A	The SMC shall have the capability of responding to system faults within a maximum of five minutes.

Table A-1. Requirements Verification Matrix (133 of 134)

Test Case ID	Requirement ID	Requirement Text
	SMC-0350#A	The SMC shall have the capability of responding to security compromises within a maximum of five minutes.
	IMS-0080#A	The IMS shall maintain a list of authorized ECS services for each user and shall update the list with information supplied by the SMC.
BC011.003	SMC-0320#A	The SMC shall be capable of scheduling ground activities to a minimum of one minute resolution.
	SMC-1300#A	The SMC shall support and maintain the ECS policies and procedures regarding instrument and ground event scheduling, including, at a minimum: a. Mission and science guidelines b. Directives for scheduling instrument data ingest, processing, reprocessing, retrieval, and data distribution
	SMC-1310#A	The SMC shall support and maintain the allocation of ground event functions and capabilities to each site and element.
	SMC-1320#A	The SMC shall support and maintain priorities used in scheduling ground events.
	SMC-1345#A	The LSM shall perform priority management services to resolve conflicts for ECS resources.
	SMC-1360#A	The SMC shall generate ground resource scheduling directives, or recommendations for FOS elements, in response to emergency situations.
BC011.004	SMC-1320#A	The SMC shall support and maintain priorities used in scheduling ground events.
	SMC-3300#A	The SMC shall monitor site and element hardware status to determine their operational states including, at a minimum: a. On-line b. Failed c. In maintenance d. In test mode e. In simulation mode
	SMC-3305#A	The LSM shall monitor its elements hardware, and scientific and system software status to determine their operational states including, at a minimum : a. On-line b. Failed c. In maintenance d. In test mode e. In simulation mode

Table A-1. Requirements Verification Matrix (134 of 134)

Test Case ID	Requirement ID	Requirement Text
BC011.005	SMC-1000#A	The SMC shall provide application programming interfaces (APIs) for the monitoring and control of managed resources. These APIs shall provide mechanisms for: a. Capturing, by an application, of management data b. Exchanging management data between a managed application and its management agent c. Exchanging management data between a management agent and the LSM d. Performing analyses and generating reports using management data
	SMC-1340#A	The SMC shall generate scheduling directives for system level, site-to-site, and element-to-element integration, testing, and simulation activities.
	SMC-2610#A	The SMC shall provide and maintain a bulletin board service with information on ECS status, events, and news.
	SMC-2620#A	The SMC shall maintain via the ECS bulletin board service, the SMC toolkit consisting of a list of SDPS approved CASE tools and references to standards for exchanging data for scientist use.
	C-MSS-10420	The MSS shall interface with the ISS subsystems to exchange the data items in Table 5.1-6 as specified in the ECS internal ICDs, 313-DV3-003.

Appendix B. Test Plan to Test Procedure Matrix

The following matrix provides a mapping of the test cases from the approved CSMS Integration and Test Plan for the ECS Project, Volume 2: Release A (319-CD-004-003) to the updated test cases in this document.

The matrix mapping the test cases from the System Integration and Test Plan for the ECS Project, Volume 2: Release A, Final (402-CD-002-002) to the test case in this document is TBD.

Table B-1. Test Plan to Test Procedure Matrix (1 of 28)

Old Case / Thread	New Case / Thread
TC036.001	TC020.012
TC036.001	TC020.012
BC012.001	BC001.001
TC019.001	TC011.001
TC036.001	TC020.012
BC023.004	TC010.002 Test case moved from build to thread
BC023.004	TC010.002 Test case moved from build to thread
BC023.004	TC010.002 Test case moved from build to thread
BC023.004	TC010.002 Test case moved from build to thread
BC023.004	TC010.002 Test case moved from build to thread
BC023.004	TC010.002 Test case moved from build to thread
BC023.004	TC010.002 Test case moved from build to thread
BC023.005	TC010.003 Test case moved from build to thread
BC023.005	TC010.003 Test case moved from build to thread
BC023.005	TC010.003 Test case moved from build to thread
BC023.005	TC010.003 Test case moved from build to thread
BC023.005	TC010.003 Test case moved from build to thread

Table B-1. Test Plan to Test Procedure Matrix (2 of 28)

Old Case / Thread	New Case / Thread
BC023.004	TC010.002 Test case moved from build to thread
BC023.005	TC010.003 Test case moved from build to thread
BC023.005	TC010.003 Test case moved from build to thread
BC023.004	TC010.002 Test case moved from build to thread
BC023.005	TC010.003 Test case moved from build to thread
BC023.005	TC010.003 Test case moved from build to thread
BC023.005	TC010.003 Test case moved from build to thread
BC023.005	TC010.003 Test case moved from build to thread
BC023.005	TC010.003 Test case moved from build to thread
BC023.005	TC010.003 Test case moved from build to thread
BC023.004	TC010.002 Test case moved from build to thread
BC023.007	BC011.010
BC023.007	BC011.010
BC031.013	BC011.008
BC023.007	BC011.010
TC019.001	TC011.001 TC011.003 TC011.004
TC019.001	TC011.001 TC011.010
TC019.001	TC011.001
TC019.001	TC011.001
TC019.001	TC011.001
TC019.001	TC011.001
TC019.003	TC011.003
TC019.001	TC011.001
TC019.001	TC011.001
TC019.001	TC011.001
TC019.001	TC011.009

Table B-1. Test Plan to Test Procedure Matrix (3 of 28)

Old Case / Thread	New Case / Thread
TC019.002	TC011.003
TC019.002	TC011.001
TC019.003	TC011.002
TC019.003	TC011.010
TC024.001	TC002.001
N/A	TC002.003
N/A	TC002.001
TC024.005	TC002.001
TC024.005	TC002.001
TC024.003	TC002.002
TC024.003	TC002.002
TC024.002	TC002.003
TC024.002	TC002.003
TC024.002	TC002.003
TC024.002	TC002.003
TC024.002	TC002.003
TC024.005	TC015.002
TC024.005	TC015.002
TC024.005	TC015.002
TC024.004	TC015.001
TC024.004	TC015.001
TC024.004	TC015.001
TC024.002	TC002.004
TC024.002	TC002.004
TC024.002	TC002.004
TC024.002	TC002.004
TC024.010	TC015.007
BC023.003	TC025.001 Test case moved from build to thread
BC023.003	TC025.004
BC023.003	TC025.001
BC023.003	TC010.001
BC023.003	TC010.001
	******
BC023.003	TC010.001 Test case moved from build to thread
BC023.003	Requirement deleted or moved to a later release

Table B-1. Test Plan to Test Procedure Matrix (4 of 28)

Old Case / Thread	New Case / Thread
BC023.003	TC025.001 TC025.002
BC023.003	TC025.001 TC025.002
BC023.003	TC025.001 TC025.002
BC023.003	TC025.001 TC025.002 TC025.005
BC023.003	TC025.001 TC025.002 TC025.004
BC023.003	TC025.001 TC025.002 TC025.004
BC023.003	TC025.003
TC022.002	TC014.002
TC022.002	TC014.002
TC022.002	TC014.002
TC022.002	TC014.002
TC022.002	TC014.002
TC022.002	TC014.002
TC022.002	TC014.002
TC022.002	TC014.002
TC022.002	TC014.002
TC022.002	TC014.002
TC022.002	TC014.002
TC022.002	TC014.002
TC022.002	TC014.002
TC022.002	TC014.002
TC022.002	TC014.002
TC022.002	TC017.001
TC022.002	TC017.001
TC022.002	TC017.001
TC022.002	TC017.001
TC022.002	TC017.001
TC022.002	TC017.001

Table B-1. Test Plan to Test Procedure Matrix (5 of 28)

Old Case / Thread	New Case / Thread
BC023.006	TC016.001 Test case moved from build to thread
BC023.006	TC016.001
BC023.006	TC016.001
BC023.006	TC016.002
BC023.006	TC016.002
BC023.006	TC016.002
BC023.006	TC016.002
BC023.006	TC016.002
BC023.006	TC016.001
BC023.006	TC016.002
BC023.006	TC016.001
TC022.002	TC014.003
TC022.002	TC014.003
TC022.002	TC014.003
TC022.002	TC014.003
TC022.002	TC014.003
TC022.002	TC014.003
N/A	TC014.003
TC022.002	TC014.003
TC022.002	TC014.003
TC028.005	TC005.001
TC028.005	TC005.001
TC028.005	TC005.001
TC028.005	TC005.001
	******
	******
	******
	******
	******
TC020.003	Moved to Rel B
TC020.003	Moved to Rel B
TC020.003	Moved to Rel B
N/A	TC013.001 TC013.002
N/A	TC013.001 TC013.002
N/A	TC013.001

Table B-1. Test Plan to Test Procedure Matrix (6 of 28)

Old Case / Thread	New Case / Thread
N/A	TC013.004
N/A	TC013.001
N/A	TC013.001
N/A	TC013.003
N/A	TC013.003
N/A	TC013.003
N/A	TC013.001
TC020.004	TC013.002
TC020.004	TC013.002
TC020.004	Moved to Release B
TC020.001	TC013.001
TC020.001	TC013.001
TC020.001	TC013.004
TC021.001	TC001.003
TC021.001	TC032.001
TC021.001	TC032.001
TC021.001	TC032.001
TC021.001	TC032.001
TC021.001	TC032.003
TC021.001	TC001.003
TC021.002	TC001.004
TC021.002	TC001.004
TC021.002	TC001.004
TC021.002	TC001.004
TC021.001	TC001.003
TC021.001	TC001.003
TC021.001	TC001.003
TC021.001	TC001.003
TC021.001	TC001.003
TC021.002	TC032.002
TC021.002	TC032.002
TC021.002	TC001.004
TC021.002	TC001.003
TC021.002	TC001.003
TC021.002	TC001.004
TC021.002	TC001.003
TC021.002	TC001.004

Table B-1. Test Plan to Test Procedure Matrix (7 of 28)

Old Case / Thread	New Case / Thread
TC021.002	TC001.004
TC021.003	TC032.003
TC021.003	TC032.003
TC021.003	TC032.003
TC021.001	TC001.003
TC021.001	TC001.003
TC021.004	TC032.004
TC021.004	TC032.004
TC021.004	TC032.004
TC021.004	TC032.004
TC030.002	TC023.002
TC021.005	TC032.005
TC021.004	TC032.004
TC021.005	TC032.005
TC021.004	TC032.004
TC021.005	TC032.005
TC021.004	TC032.004
TC021.005	TC032.005
TC021.005	TC032.005
TC021.005	TC032.005
TC021.005	TC032.005
TC021.005	TC032.005
TC021.005	TC032.005
TC021.005	TC032.005
TC021.005	TC032.005
TC021.005	TC032.005
TC021.006	TC032.006
TC021.005	TC032.005
TC021.006	TC032.006
TC021.006	TC032.006
TC018.001	TC033.001
TC018.001	TC033.001
TC018.001	TC033.001
TC018.001	TC033.001
TC018.002	TC033.002
TC032.001	BC010.015
TC032.001	BC010.015

Table B-1. Test Plan to Test Procedure Matrix (8 of 28)

Old Case / Thread	New Case / Thread
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.001	BC010.015
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016

Table B-1. Test Plan to Test Procedure Matrix (9 of 28)

Old Case / Thread	New Case / Thread
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.002	BC010.016
TC032.003	BC010.009
TC032.003	BC010.009
TC032.003	BC010.009
TC032.003	BC010.009
TC032.003	BC010.009
TC032.003	BC010.009
TC032.003	BC010.009
TC032.003	BC010.009
TC032.003	BC010.009
TC032.003	BC010.009
TC032.003	BC010.009
TC032.003	BC010.009
TC032.003	BC010.009
TC032.003	BC010.009
TC032.003	BC010.009
TC032.001	BC010.015
TC032.002	BC010.016
TC032.001	BC010.015
TC032.002	BC010.016
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012

Table B-1. Test Plan to Test Procedure Matrix (10 of 28)

Old Case / Thread	New Case / Thread
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.001	BC010.012
TC033.002	BC010.013
TC033.002	BC010.013
TC033.002	BC010.013
TC033.002	BC010.013
TC033.002	BC010.013
TC033.002	BC010.013
TC033.002	BC010.013
TC033.002	BC010.013
TC033.002	BC010.013
TC033.002	BC010.013
TC033.002	BC010.013
TC033.003	BC010.013
TC033.002	BC010.013
TC033.003	BC010.013
TC033.002	BC010.013
TC033.002	BC010.013

Table B-1. Test Plan to Test Procedure Matrix (11 of 28)

Old Case / Thread	New Case / Thread
TC033.002	BC010.013
TC033.002	BC010.013
TC033.002	BC010.013
TC033.002	BC010.013
TC033.002	BC010.013
TC033.002	BC010.013
TC033.002	BC010.013
TC033.002	BC010.013
TC033.002	BC010.013
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.003	BC010.014
TC033.001 TC033.002	BC010.012 BC010.013
TC033.001	BC010.012
TC033.002	BC010.013
TC033.001 TC033.002	BC010.012 BC010.013

Table B-1. Test Plan to Test Procedure Matrix (12 of 28)

Old Case / Thread	New Case / Thread
TC033.001 TC033.002	BC010.013
TC033.001	BC010.012
TC033.001	BC010.012
TC033.002	BC010.013
TC033.002	BC010.013
TC033.003	BC010.014
TC034.001	BC010.004
TC034.001	BC010.004
TC034.001	BC010.004
TC034.001	BC010.004
TC034.001	BC010.004
TC034.002	BC010.005
TC034.002	BC010.005
TC034.002	BC010.005
TC034.002	BC010.005
TC034.002	BC010.005
TC034.002	BC010.005
TC034.002	BC010.005
TC034.002	BC010.005
TC034.002	BC010.005
TC034.002	BC010.005
TC034.002	BC010.005
TC034.003	BC010.006
TC034.003	BC010.006
TC034.003	BC010.006
TC035.001	BC010.012
TC035.001	BC010.012
TC035.001	BC010.014
TC035.001	Requirements will be tested via inspection.
TC035.001	" "
TC035.001	" "
TC035.001	" "
TC035.002	" "
TC035.002	" "
TC035.002	" "
TC035.002	" "

Table B-1. Test Plan to Test Procedure Matrix (13 of 28)

Old Case / Thread	New Case / Thread
TC035.002	" "
TC035.002	" "
TC035.002	" "
TC035.002	" "
TC035.002	" "
TC035.002	" "
TC035.002	" "
TC035.002	" "
TC035.002	" "
TC035.002	" "
TC035.003	" "
TC035.003	" "
TC035.002	" "
TC035.002	" "
TC035.002	" "
TC035.002	" "
TC035.003	" "
TC035.002	" "
TC035.002	Requirements will be tested via inspection.
TC035.002	" "
TC035.002	" "
TC035.002	" "
TC035.003	" "
TC035.002	" "
TC035.002	" "
TC035.002	" "
TC035.002	" "
TC035.002	" "
TC017.003	BC010.004 BC010.005
TC017.001	TC001.001
TC017.001	TC001.001
TC017.003	BC010.004 BC010.005
TC017.003	BC010.004 BC010.005
TC017.003	BC010.004 BC010.005

Table B-1. Test Plan to Test Procedure Matrix (14 of 28)

Old Case / Thread	New Case / Thread
TC017.003	BC010.004 BC010.005
TC017.003	BC010.004 BC010.005
TC017.003	BC010.004 BC010.005
TC017.003	BC010.004 BC010.005
TC017.003	BC010.004 BC010.005
TC017.003	BC010.004 BC010.005
TC017.003	BC010.004 BC010.005
TC017.003 TC017.004	BC010.004 BC010.005
TC017.004	BC010.004 BC010.005
TC017.004	BC010.004 BC010.005
TC017.004	BC010.004 BC010.005
TC017.004	BC010.004 BC010.005
TC017.003 TC017.004	BC010.004 BC010.005
TC017.003 TC017.004	BC010.004 BC010.005
TC017.003 TC017.004	BC010.004 BC010.005
TC017.003 TC017.004	BC010.004 BC010.005
TC017.003 TC017.004	BC010.004 BC010.005
TC017.003 TC017.004	BC010.004 BC010.005
TC017.003 TC017.004	BC010.004 BC010.005
TC017.003 TC017.004	BC010.004 BC010.005

Table B-1. Test Plan to Test Procedure Matrix (15 of 28)

Old Case / Thread	New Case / Thread
TC017.003 TC017.004	BC010.004 BC010.005
TC017.003 TC017.004	BC010.004 BC010.005
TC017.003 TC017.004	BC010.004 BC010.005
TC017.003 TC017.004	BC010.004 BC010.005
TC017.003 TC017.004	BC010.004 BC010.005
TC017.003 TC017.004	BC010.004 BC010.005
TC017.001	TC001.001
TC017.001	TC001.002
TC017.001	TC001.001
TC017.001	TC001.001
TC017.001	TC001.002
TC017.004	TC001.004
TC017.004	TC001.004
TC017.005	TC001.005
TC017.004	TC001.004 TC026.001
TC017.004	TC001.004
TC017.004	TC001.004
TC017.003 TC017.004 TC017.005 TC028.010	TC001.003 TC001.004 TC001.005 TC020.010
TC017.004	TC001.004
TC028.10	
TC017.004	TC001.004
TC017.004 TC028.010	TC001.004 TC020.010
TC017.003 TC017.004 TC017.005 TC028.010	TC001.003 TC001.004 TC001.005 TC020.010
TC017.003 - TC017.005 TC028.010	TC001.003 TC001.005 TC020.010

Table B-1. Test Plan to Test Procedure Matrix (16 of 28)

Old Case / Thread	New Case / Thread
TC017.003 - TC017.005 TC028.010	TC001.003 TC001.005 TC020.010
TC017.003 - TC017.005 TC028.010	TC001.003 TC001.005 TC020.010
TC017.003 - TC017.005 TC028.010	TC001.003 TC001.005 TC020.010
TC017.003 - TC017.005 TC028.010	TC001.003 TC001.005 TC020.010
TC017.003 - TC017.005 TC028.010	TC001.003 TC001.005 TC020.010
TC017.003 - TC017.005 TC028.010	TC001.003 TC001.005 TC020.010
TC017.003 - TC017.005 TC028.010	TC001.003 TC001.005 TC020.010
TC017.003 - TC017.005 TC028.010	TC001.003 TC001.005 TC020.010
TC017.003 - TC017.005 TC028.010	TC001.003 TC001.005 TC020.010
TC017.003 - TC017.005 TC028.010	TC001.003 TC001.005 TC020.010
TC017.003 - TC017.005 TC028.010	TC001.003 TC001.005 TC020.010
TC017.003 - TC017.005 TC028.010	TC001.003 TC001.005 TC020.010

Table B-1. Test Plan to Test Procedure Matrix (17 of 28)

Old Case / Thread	New Case / Thread
TC036.001	TC020.012
TC036.001	TC020.012
TC036.001	TC020.012
TC036.001	TC020.012
BC031.012	BC011.009
BC031.012	BC011.009
BC031.012	BC011.009
BC031.012	BC011.009
BC031.012	BC011.009
BC031.012	BC011.009
BC031.012	BC011.009
BC031.012	BC011.009
BC031.012	BC011.009
BC031.012	BC011.009
BC031.012	BC011.009
BC031.013	BC011.008
BC031.013	BC011.008
BC031.013	BC011.008
BC031.013	BC011.008
BC031.013	BC011.008
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003

Table B-1. Test Plan to Test Procedure Matrix (18 of 28)

Old Case / Thread	New Case / Thread
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC028.009	TC006.002 TC020.001, TC020.002, TC020.003
TC028.009	TC006.002 TC020.001, TC020.002, TC020.003
TC028.009	TC006.002 TC020.001, TC020.002, TC020.003
TC028.009	TC006.002 TC020.001, TC020.002, TC020.003
TC028.009	TC006.002 TC020.001, TC020.002, TC020.003
TC028.009	TC006.002 TC020.001, TC020.002, TC020.003
TC028.009	TC006.002 TC020.001, TC020.002, TC020.003
TC028.009	TC006.002 TC020.001, TC020.002, TC020.003
TC028.009	TC006.002 TC020.001, TC020.002, TC020.003
TC028.009	TC006.002 TC020.001, TC020.002, TC020.003
TC028.009	TC006.002 TC020.001, TC020.002, TC020.003
TC028.009	TC006.002 TC020.001, TC020.002, TC020.003

Table B-1. Test Plan to Test Procedure Matrix (19 of 28)

Old Case / Thread	New Case / Thread
TC028.009	TC006.002 TC020.001, TC020.002, TC020.003
BC031.009	TC022.001
BC031.009	TC022.001
BC031.009	TC022.001
BC031.009	TC022.001
BC031.009	TC022.001
BC031.009	TC022.001
BC031.009	TC030.001
BC031.009	TC022.001
BC031.009	TC022.001
BC031.009	TC005.001
BC031.009	TC022.001
BC031.009	TC005.001
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC028.009	TC006.001 TC020.001, TC020.002, TC020.003
TC029.007	TC003.001 TC004.001 TC019.001 TC019.002
TC029.007	TC003.001 TC004.001 TC019.001 TC019.002
TC029.007	TC004.001 TC019.002
TC029.007	TC004.001 TC019.002
TC029.007	TC004.001 TC019.002
TC029.007	TC019.001
TC029.007	TC003.001
TC029.007	TC004.001 TC019.002

Table B-1. Test Plan to Test Procedure Matrix (20 of 28)

Old Case / Thread	New Case / Thread
TC029.007	TC019.001
TC029.007	TC019.001
TC027	TC008.001 TC031.001 TC031.005
TC0027	TC008.001 TC031.001
TC027.005	TC008.001 TC031.001
TC027.005	TC008.001 TC031.001
TC027.005	TC008.001 TC031.001
TC027.005	TC008.001 TC031.001
TC027.005	TC008.001 TC031.001 TC039.001
TC027.005	TC008.001 TC031.001 TC039.001
TC027.004	TC039.001 TC031.005
TC027.005	TC008.001 TC031.001
TC027.007	TC008.001 TC039.003
TC027.007	TC039.001 TC039.002
TC027.007	TC039.001
TC027.007	TC039.001
TC027.007	TC039.001 TC039.002
TC027.007	TC039.001 TC039.002
TC027.007	TC039.001 TC039.002
TC027.007	TC039.001
TC027.007	TC039.001
TC027.007	TC039.001
TC027.007	TC039.001

Table B-1. Test Plan to Test Procedure Matrix (21 of 28)

Old Case / Thread	New Case / Thread
TC027.007	TC039.001 TC039.002
TC027.007	TC039.001 TC039.002
TC027.007	TC039.001 TC039.002
TC027.006	TC008.004 TC031.004
TC027.006	TC031.003
TC027.006	TC039.001 TC031.003
TC027.001	TC039.001 TC008.001 TC008.002
TC027.001	TC039.001 TC008
TC027.001	TC039.001 TC008.001
TC027.006	TC039.001 TC031.003
TC027.001	TC039.001
TC027.001	TC039.001 TC008.003
TC027.001	TC039.001 TC008.004
TC027.001	TC039.001 TC008.004
TC027.001	TC039.001 TC008
TC027.001	TC039.001 TC008.004
TC027.001	TC039.001 management services 2 build
TC027.006	TC008.004
TC027.001	TC039.001 TC008.004
TC027.001	TC039.001 TC008.004
TC027.001	TC039.001 TC008.004

Table B-1. Test Plan to Test Procedure Matrix (22 of 28)

Old Case / Thread	New Case / Thread
TC027.001	TC039.001 TC008.004
TC027.003	TC039.003
TC027.003	TC039.003 TC009
TC027.003	TC039.003 TC009
TC027.003	TC039.001 SMC change request
TC027.003	TC039.001 TC008.003 TC039.002
TC027.003	TC039.001 TC008.002 Moved to SMC CM
TC027.003	TC039.001 TC008.002 Moved to SMC CM
TC027.003	TC039.001 TC008.002 Moved to SMC CM
TC027.003	TC039.001 TC008.003 Moved to SMC CM
TC027.003	TC039.001 TC008.003 Moved to SMC CM
TC027.003	TC039.003 TC039.001 TC008.003
TC027.003	TC039.001 TC008.002 Moved to SMC CM
TC027.003	TC039.003 TC008.003
TC027.004	TC008.001 TC039.002
TC027.006	TC008.004 TC031.002
TC027.006	TC008.004 TC031.002
N/A	TC007.009

Table B-1. Test Plan to Test Procedure Matrix (23 of 28)

Old Case / Thread	New Case / Thread
N/A	TC007.001
N/A	TC007.010
N/A	TC007.002
N/A	TC007.003
N/A	TC007.004
N/A	TC007.005
N/A	TC007.006
N/A	TC017.001
N/A	TC007.009
N/A	TC007.007
N/A	TC007.007
N/A	TC007.008
TC029.001	TC021.001
TC029.001	TC027.001
TC029.001	TC027.001
TC029.001	TC027.001
TC029.001	TC027.001
TC029.001	TC027.001
TC029.001	TC027.001
TC029.001	TC027.002
TC029.002	TC027.002
TC029.002	TC027.003
TC029.002	TC027.002
TC029.002	TC027.002
TC029.001	TC027.002
TC029.002	TC027.003
TC029.002	TC027.003
TC029.002	TC027.003
TC029.002	TC027.003
TC029.002	TC027.001
TC029.002	TC027.004
TC029.002	TC027.004
TC029.002	TC027.004
TC029.002	TC027.004
TC029.003	TC021.003
TC029.003	TC021.003
TC029.003	TC021.003

Table B-1. Test Plan to Test Procedure Matrix (24 of 28)

Old Case / Thread	New Case / Thread
TC029.003	TC021.003
TC029.003	TC021.003
TC029.003	TC021.003
TC029.003 (test case needs clarification)	TC021.003
TC029.0030	TC021.003
TC029.003	TC021.003
TC029.003	TC021.003
TC029.004	TC027.004
TC029.004	TC027.003
TC029.004	TC021.004
TC029.004	TC021.004
TC029.002	TC021.005
TC029.005	TC021.005
TC029.005	TC021.005
TC029.005	TC021.005
TC029.005	TC021.005
TC029.006	TC027.002
TC029.006	TC027.002
TC029.006	TC027.002
TC028.006	TC020.009
TC028.006	TC020.009
TC028.006	TC020.009
TC028.006	TC020.009
TC028.006	TC020.009
TC028.006	TC020.009
TC028.001	TC020.004
TC028.001	TC020.004
TC028.002	TC020.005
TC028.003	TC020.006
TC028.002	TC020.005
TC028.001	TC020.004
TC028.005	TC020.008
TC028.008	TC020.011 TC030.001 TC030.002

Table B-1. Test Plan to Test Procedure Matrix (25 of 28)

Old Case / Thread	New Case / Thread
TC028.008	TC020.011 TC030.001 TC030.002
TC028.005	TC020.008
TC028.005	TC020.008
TC028.005	TC020.008
TC028.005	TC020.008
TC028.005	TC020.008
TC028.004	TC020.007
TC028.004	TC020.007
TC028.004	TC020.007
TC028.004	TC020.007
TC028.004	TC020.007
TC028.004	TC020.007
TC028.005	TC020.008
TC028.005	TC020.008
TC028.005	TC020.008
TC028.005	TC020.008 TC030.001 TC030.002
TC028.005	TC020.008
TC028.005	TC020.008
TC028.005	TC020.008
TC028.004	TC020.007 TC030.001 TC030.002
TC028.006	TC020.009 TC030.001 TC030.002
TC028.006	TC020.009
TC028.008	TC020.0011
TC028.008	TC020.011
TC028.008	TC020.011
TC028.008	TC020.011
TC028.008	TC020.011
TC028.008	TC020.011
TC028.008	TC020.011
TC028.008	TC020.011
TC028.008	TC020.011

Table B-1. Test Plan to Test Procedure Matrix (26 of 28)

Old Case / Thread	New Case / Thread
TC028.008	TC020.011
TC028.007	TC020.010
TC028.007	TC020.010
TC028.007	TC020.010
TC028.007	TC020.010
TC024.006	TC015.003
TC024.002	TC028.001
TC025.002	TC028.001
TC024.006	TC028.006
TC024.006	TC028.002
TC024.006	TC028.002
TC024.006	TC028.002
TC024.014	TC028.002
TC024.014	TC028.002
TC024.014	TC028.002
TC024.014	TC028.002
TC024.014	TC028.002
TC024.014	TC028.003
TC024.006	TC028.003
TC024.006	TC028.003
TC024.006	TC028.003
TC024.006	TC028.003
TC024.014	TC028.004
TC024.006	TC028.004
BC031.010	TC028.004
TC024.014	TC028.004
TC024.011	TC028.005
TC024.011	TC028.005
TC024.011	TC028.005
TC030.003 TC030.005 TC030.006	TC023.003 TC023.005 TC023.006
TC030.004	TC023.004
TC030.005	TC023.005
TC030.005	TC023.005
TC030.006	TC023.006
TC030.005	TC023.005

Table B-1. Test Plan to Test Procedure Matrix (27 of 28)

Old Case / Thread	New Case / Thread
TC030.005	TC023.005 TC028.003
TC030.006	TC023.006
TC030.006	TC023.006
TC030.006	TC023.006
TC030.006	TC023.006
TC030.006	TC023.006
TC030.006	TC023.006
TC030.006	TC023.006
TC030.006	TC023.006
TC030.006	TC023.006
TC030.006	TC023.006
TC027.006	TC009.003
N/A	TC009.001
TC027.006	TC009.003
TC027.006	TC009.003
TC027.006	TC009.003
TC027.006	TC009.003
N/A	TC009.001
N/A	TC009.001
TC027.006	TC009.003
TC027.006	TC009.003
TC027.006	TC009.002 TC009.003
TC027.006	TC009.002 TC009.003
N/A	TC009.001
N/A	TC009.001
N/A	TC009.001
BC031.008	BC009.010
BC031.008	BC009.010
BC031.008	BC009.010
BC031.008	BC009.010
BC031.008	BC009.010
BC031.008	BC009.010
BC031.008	BC009.010
BC031.008	BC009.010
BC031.008	BC009.010

Table B-1. Test Plan to Test Procedure Matrix (28 of 28)

Old Case / Thread	New Case / Thread
BC031.008	BC009.010
BC031.008	BC009.010
BC031.008	BC009.010
BC031.008	BC009.010
BC031.008	BC009.010
BC031.008	BC009.010
BC031.008	BC009.010
BC031.008	BC009.010
BC031.008	BC009.010
BC031.008	BC009.010
BC031.011	BC009.011
BC031.011	BC009.011
BC031.011	BC009.011
BC031.011	BC009.011
BC031.011	BC009.011
BC031.011	BC009.011
BC031.010	BC009.011
BC031.010	TC029.001
BC031.010	TC029.001

Abbreviations and Acronyms

ACL	access control list
AI&T	Algorithm Integration and Test
API	application programming interface
ATM	asynchronous transfer mode
BBS	bulletin board server
CCB	configuration control board
CCR	configuration change request
CDR	Critical Design Review
CDRL	contract data requirements list
CDS	Cell Directory Service
CDSCP	cell directory service command program
CERES	Clouds and Earth's Radiant Energy System
CI	configuration item
CM	configuration management
CMAS	Configuration Management Application Service
CORBA	common object request broker architecture
COTS	commercial off-the-shelf
CRM	change request manager
CSC	computer software component
CSCI	computer software configuration item
CSMS	Communications and Systems Management Segment
CSR	Consent to Ship Review
CSS	Communications SubSystem
CSU	computer software unit
DAAC	Distributed Active Archive Center
DBMS	Data Base Management System

DCCI	Distributed Computing CI
DCE	Distributed Computing Environment
DCHCI	Distributed Computing Hardware CI
DCN	document change notice
DDTS	Distributed Defect Tracking System
DFD	data flow diagram
DFS	distributed file service
DID	data item description
DNS	Domain Name Service
DOF	distributed object framework
DTS	distributed time service
DV1	document version 1
Ecom	EOS Communications
ECS	EOSDIS Core System
EDC	EROS Data Center (DAAC)
EDF	ECS Development Facility
EDHS	ECS Data Handling System
EDOS	EOS Data and Operations System
EMC	enterprise management center
EOC	ECS Operations Center
EOS	Earth Observation System
EOSDIS	Earth Observation System Data Information System
EROS	Earth Resources Observation System
ESN	EOSDIS Science Network
ETR	Element Test Review
F&PRS	Functional and Performance Requirements Specifications
FDF	flight dynamics facility
FDDI	fiber distributed data interface
FMAS	fault management application service

FOS	Flight Operations Segment
FTP	file transfer protocol
GDS	Global Directory Service
GSFC	Goddard Space Flight Center
HiPPI	High Performance Parallel Interface
HWCI	Hardware CI
I&T	Integration and Test
IATO	Independent Acceptance Test Organization
ICD	Interface Control Document
IDL	interface definition language
IDR	internal design review
INCI	Internetworking CI
INHCI	Internetworking Hardware CI
IP	Internet protocol
IR-1	Interim Release one
ISO	International Standards Organization
ISS	Internetworking SubSystem
IST	Instrument Support Terminal
IV&V	Independent Verification and Validation
JPL	Jet Propulsion Laboratory
LAN	Local Area Network
LaRC	Langley Research Center
LIS	Lightning Imaging Sensor
LSM	local systems management
M&O	Maintenance and Operations
MACI	Management Agent Software CI
MCI	Management Software CI
MG1	Management one
MHCI	Management Hardware CI

MLCI	Management Logistics Software CI
MOC	Mission Operations Center
MRS	Malfunctions/failure Reports
MSFC	Marshall Space Flight Center
MSS	Management SubSystem
MUI	management user interface
NASCOM	NASA Communications
NISS	NASA Institutional Support System
NNTP	Network News Transfer Protocol
NOAA	National Oceanic and Atmospheric Administrator
NOLAN	Nascom Operational Local Area Network
NRCA	Noncomformance Reporting And Corrective Action
NSI	NASA Science Internet
OODCE	Object Oriented DCE
OMT	Object Modeling Technique
OSI	Open System Interconnect
OSPF	Open Shortest Path First (routing protocol)
OSI-RM	OSI - Reference Model
PDR	preliminary design review
PGS	Product Generation System
PI	Primary Investigator
PMAS	Performance Management Application Service
PSCN	Program Support Communications Network
RFC	request for comment
RFP	Request for Proposal
RIP	Router Information Protocol
RMA	Reliability, Maintainability, and Availability
RPC	remote procedure call
RRR	Release Readiness Review

RTM	requirements traceability matrix
SCF	Science Computing Facility
SDPF	Science Data Processing Facility
SDPS	Science Data Processing Segment
SEI	Software Engineering Institute
SMC	systems management center (ECS)
SMTP	Simple Mail Transport Protocol
SI&T	Systems Integration and Test
SLOC	Suggested Lines of Code
STD	state transition diagram
StP	Software Through Pictures
TBD	to be determined
TCP	Transmission Control Protocol
TMI	TRMM Microwave Image
TRMM	Tropical Rainfall Measuring Mission (joint US-Japan)
TRR	test readiness review
TSDIS	TRMM Science Data and Information System
UTC	universal time code
VIRS	Visisble Infrared Scanner
V0	Version Zero
VOB	Version Object Base

This page intentionally left blank.