

4.5 Interoperability Subsystem Overview

The Interoperability Subsystem (IOS) or Advertising Service allows SDPS servers and non-ECS users to insert and subsequently search for Earth Science related services, advertisement providers, and data.

The Advertising Service provides interfaces for supporting browsing, searching, and retrieving of advertisements. Although there is a single correct format for submitting advertisements to the Advertising Service, different interfaces support database searching, text searching, and hyper-linked data access and retrieval according to the viewing styles such as plain ASCII text, interactive form, or HTML document.

There are two types of advertisements: service and product. Each type is associated with the provider submitting the advertisement. Each type also has sub-types. For example, the Science Data Server submits signature service advertisements, which are a type of service advertisement.

The SDSRV CSCI and non-ECS users advertise data collections and services with the Advertising Service by adding ESDTs. An advertisement describes the data collection with a set of product attributes. Signature services related to an ESDT are also advertised. Signature services, such as acquire, contain a signature and a server UR needed to retrieve granule data from the Data Server. The signature is parsed by a client application to determine what parameters should be passed to the server for this service. In addition, the Subscription Server advertises subscription events such as those registered by the Science Data Server.

Product advertisements include collection level metadata and therefore, the attributes reflected in the Advertising Service include a subset of SDPS Core Metadata collection level attributes.

The advertisements are stored in a relational database and the Advertising Server provides a COTs interface to the database.

Interoperability Subsystem Context

Figure 4.5-1 is the Interoperability Subsystem context diagram. The diagram shows the events sent to the Interoperability Subsystem and the events the Interoperability Subsystem sends to other subsystems. Table 4.5-1 provides descriptions of the interface events shown in the Interoperability Subsystem context diagram.

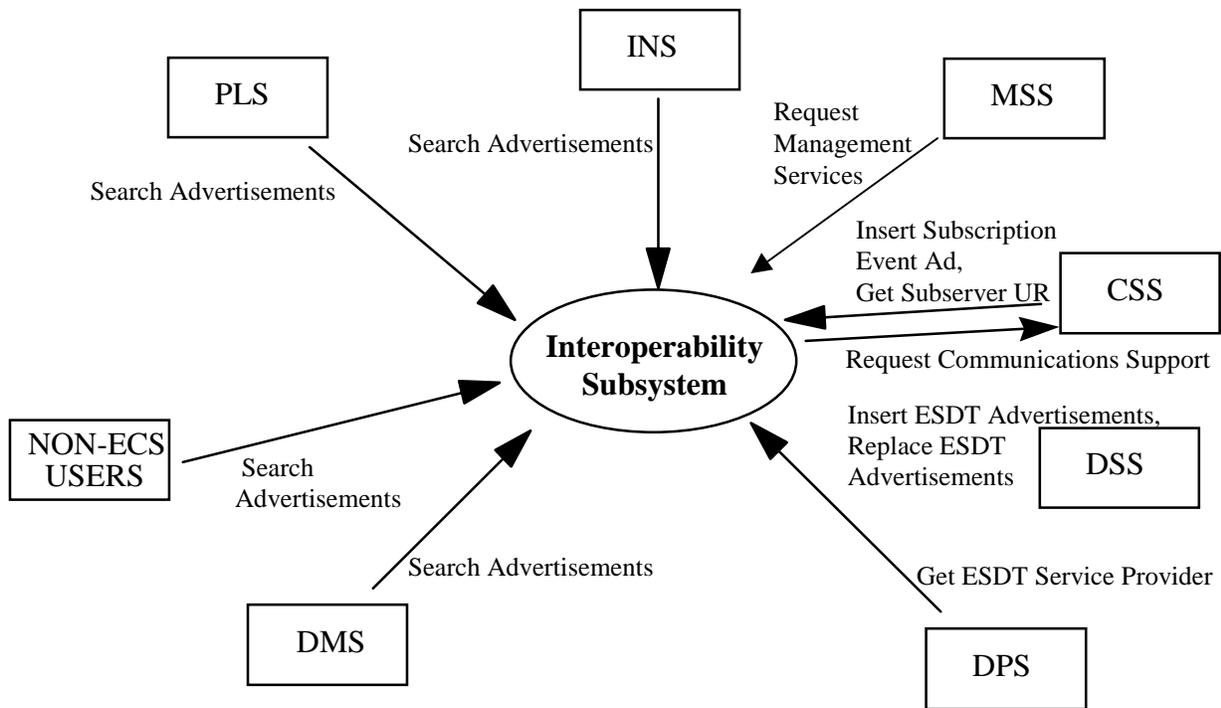


Figure 4.5-1. Interoperability Subsystem Context Diagram

Table 4.5-1. Interoperability Subsystem Interface Events (1 of 2)

Event	Interface Event Description
Search Advertisements	The IOS receives requests to search for subscription event and signature service advertisements from the PLS, DMS, and INS. These subsystems enter subscriptions with the CSS Subscription Server or obtain the proper signatures for acquiring data granules from the DSS (for the insert and update of metadata within the DSS). Non-ECS users also search for advertisements, which are essentially directory searches for the types of data that exist in the system.
Request Management Services	The MSS provides a basic management library of services to the subsystems, implemented as client or server applications, using the CSS Process Framework. The basic management library of services include: <ul style="list-style-type: none"> • Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training).
Insert Subscription Event Ad	The IOS receives requests to insert subscription event advertisements from the CSS Subscription Server.
Get Subserver UR	The CSS submits a request to the IOS to retrieve the correct subscription server UR.

Table 4.5-1. Interoperability Subsystem Interface Events (2 of 2)

Event	Interface Event Description
Request Communications Support	The CSS provides a library of services available to each SDPS and CSMS subsystem. The services required to perform the specific subsystem assignments are requested by the subsystem from the CSS. These services include: DCE support, file transfer services, Network & Distributed File Services, Bulk Data transfer services, file copying services, name/address services, password services, Server Request Framework (SRF), UR, Error/Event logging, message passing, Fault Handling services, User authentication services, Mode information, and retrieving the requested configuration attribute-value pairs from the Configuration Registry for ECS applications that request them.
Insert ESDT Advertisements	The IOS receives requests to insert advertisements for data types (ESDTs) from the DSS that includes both data product and signature service advertisements.
Replace ESDT Advertisements	The Interoperability Subsystem (IOS) receives requests to update advertisements for data types (ESDTs) from the DSS including both data product and signature service advertisements. These updated advertisements replace the previous advertisement associated with this ESDT.
Get ESDT Service Provider	The DPS sends search requests for signature service advertisements to the IOS. The DPS obtains the proper signatures and universal references for communicating with the DSS.

Interoperability Subsystem Structure

The IOS is one CSCI, ADSRV, and one HWCI, the Interface Hardware CI. The Interface Hardware CI is shared with the Data Management Subsystem

- The Advertising Service (EcIoAdServer) is a software configuration item. The Advertising Service manages Earth Science related advertisements. The advertisement information is stored persistently in a relational Database Management System (DBMS). The Advertising Service data is replicated within each DAAC using Sybase COTS software.
- The Interoperability Subsystem contains one hardware CI, the Interface Hardware (INTHW) co-owned by the Data Management Subsystem hardware. The INTHW CI provides processing and storage for the ADSRV (Advertising Service) software configuration item.

Use of COTS in the Interoperability Subsystem

- RogueWave's Tools.h++

The Tools.h++ class libraries are used by the IOS to provide basic functions and objects such as strings and collections. These libraries must be installed with the IOS software for any of the IOS processes to run.

- RogueWave's DBTools.h++

The DBTools.h++ C++ class libraries are used to interact with the Sybase database SQL server. The use of DBTools buffers the IOS processes from the relational database used. These libraries must be installed with the IOS for the Advertising Server to run and allow client processes to perform queries of Advertising database information.

- Sybase Server

Sybase's SQL server provides access for the Advertising Service to insert, update and delete advertisement database information. The Sybase SQL Server must be running during operations for the Advertising Server to execute search and update requests on the Advertisement database.

- Netscape Enterprise Server

Netscape's Enterprise server is used by the Advertising Service CSCI for interpreting Hypertext Transport Protocol (HTTP) allowing users to search, insert, and maintain advertisements. Hypertext Markup Language (HTML) web pages are included as part of the Advertising Service CSCI to allow access via the Netscape Enterprise Servers at the DAACs to make requests.

4.5.1 Advertising Service Software Description

4.5.1.1 Advertising Service Functional Overview

The Advertising Service (ADSRV) CSCI is two processes, the Advertising Server and the Earth Science On-line Directory (ESOD).

The Advertising Server is a background process that interacts with the Advertising persistent store for searching, inserting and updating advertisements.

The ESOD is a combination of HTML web pages and CGI programs called from the HTML web pages to communicate with the Advertising Server. The web pages provide an interface to allow users to:

- Search for Advertisements: Users can search for Earth Science related data and services through the web interfaces of the Earth Science On-line Directory. Searches are done with specific attributes or with wild card text.

4.5.1.2 Advertising Service Context

Figure 4.5.1.2-1 is the Advertising Service CSCI context diagram. The diagram shows the events sent to the ADSRV CSCI and the events the ADSRV CSCI sends to other CSCIs. Table 4.5.1.2-1 provides descriptions of the interface events shown in the ADSRV CSCI context diagram.

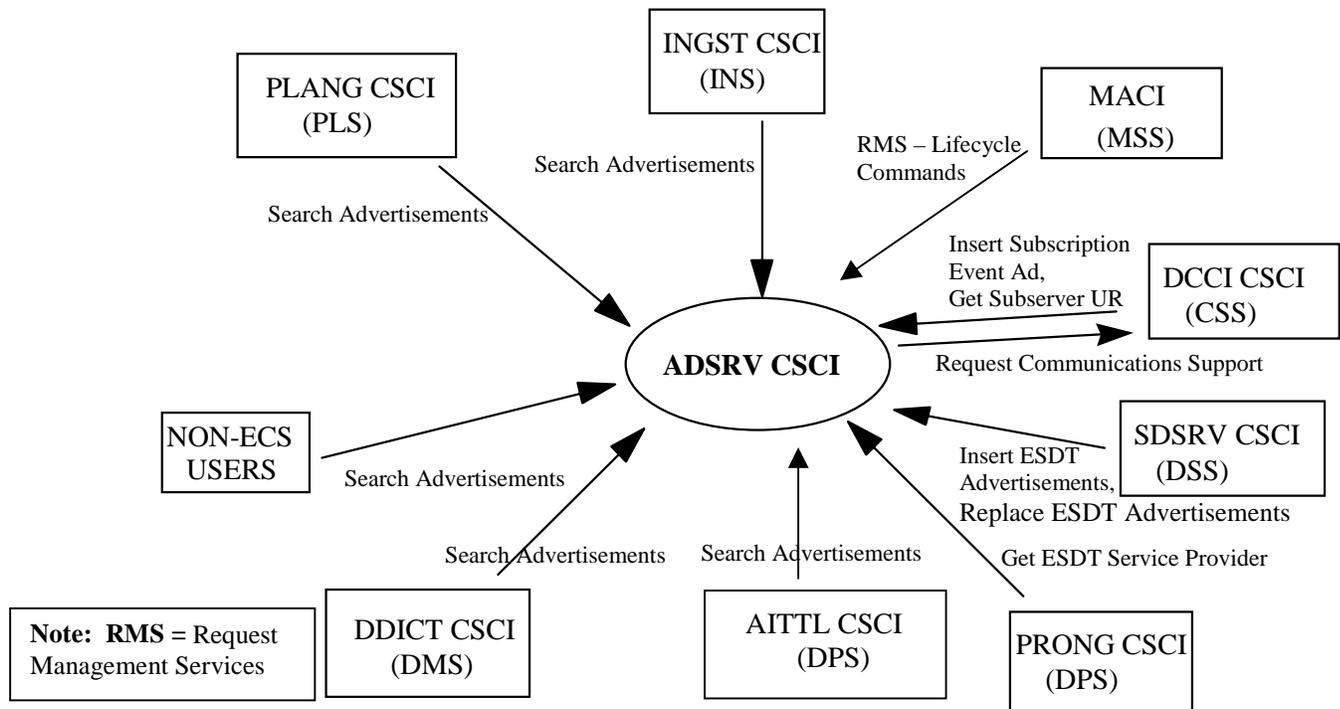


Figure 4.5.1.2-1. Advertising Service CSCI Context Diagram

Table 4.5.1.2-1. Advertising Service CSCI Interface Events (1 of 2)

Event	Interface Event Description
Search Advertisements	The ADRSV CSCI receives requests to search for subscription event and signature service advertisements from the PLANG, DDICT, PRONG, AITTL and INGST CSCIs. These CSCIs enter subscriptions with the DCCI CSCI Subscription Server or obtain the proper signatures for acquiring data granules from the SDRSV CSCI (for the insert and update of metadata within the SDRSV archives). Non-ECS users also search for advertisements, which are essentially directory searches for the types of data that exist in the system.
Request management services	The MACI provides a basic management library of services to the CSCIs, implemented as client or server applications, using the CSS Process Framework. The basic management library of services include: <ul style="list-style-type: none"> • Lifecycle Commands – The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training).

Table 4.5.1.2-1. Advertising Service CSCI Interface Events (2 of 2)

Event	Interface Event Description
Insert Subscription Event Ad	The ADSRV CSCI receives requests to insert subscription event advertisements from the DCCI CSCI Subscription Server.
Get Subserver UR	The DCCI CSCI sends a request to the ADSRV CSCI to retrieve the correct subscription server UR.
Request Communications Support	The DCCI CSCI provides a library of services available to each SDPS and CSMS CSCI. The services required to perform the specific CSCI assignments are requested by the CSCI from the DCCI CSCI. These services include: DCE support, file transfer services, Network & Distributed File Services, Bulk Data transfer services, file copying services, name/address services, password services, Server Request Framework (SRF), UR, Error/Event logging, message passing, Fault Handling services, User authentication services, Mode information, and retrieving the requested configuration attribute-value pairs from the Configuration Registry for ECS applications that request them.
Insert ESDT Advertisements	The ADSRV CSCI receives requests to insert advertisements for data types (ESDTs) from the SDSRV CSCI that includes both data product and signature service advertisements.
Replace ESDT Advertisements	The ADSRV CSCI receives requests to insert updated advertisements for data types (ESDTs) from the SDSRV CSCI including both data product and signature service advertisements. This information replaces the original information.
Get ESDT Service Provider	The PRONG CSCI sends search requests for signature service advertisements to the ADSRV CSCI. The PRONG CSCI obtains the proper signatures and universal references for communicating with the SDSRV CSCI.

4.5.1.3 Advertising Service Architecture

Figure 4.5.1.3-1 is the ADSRV CSCI architecture diagram. The diagram shows the events sent to the ADSRV CSCI processes and the events the ADSRV CSCI processes send to other processes.

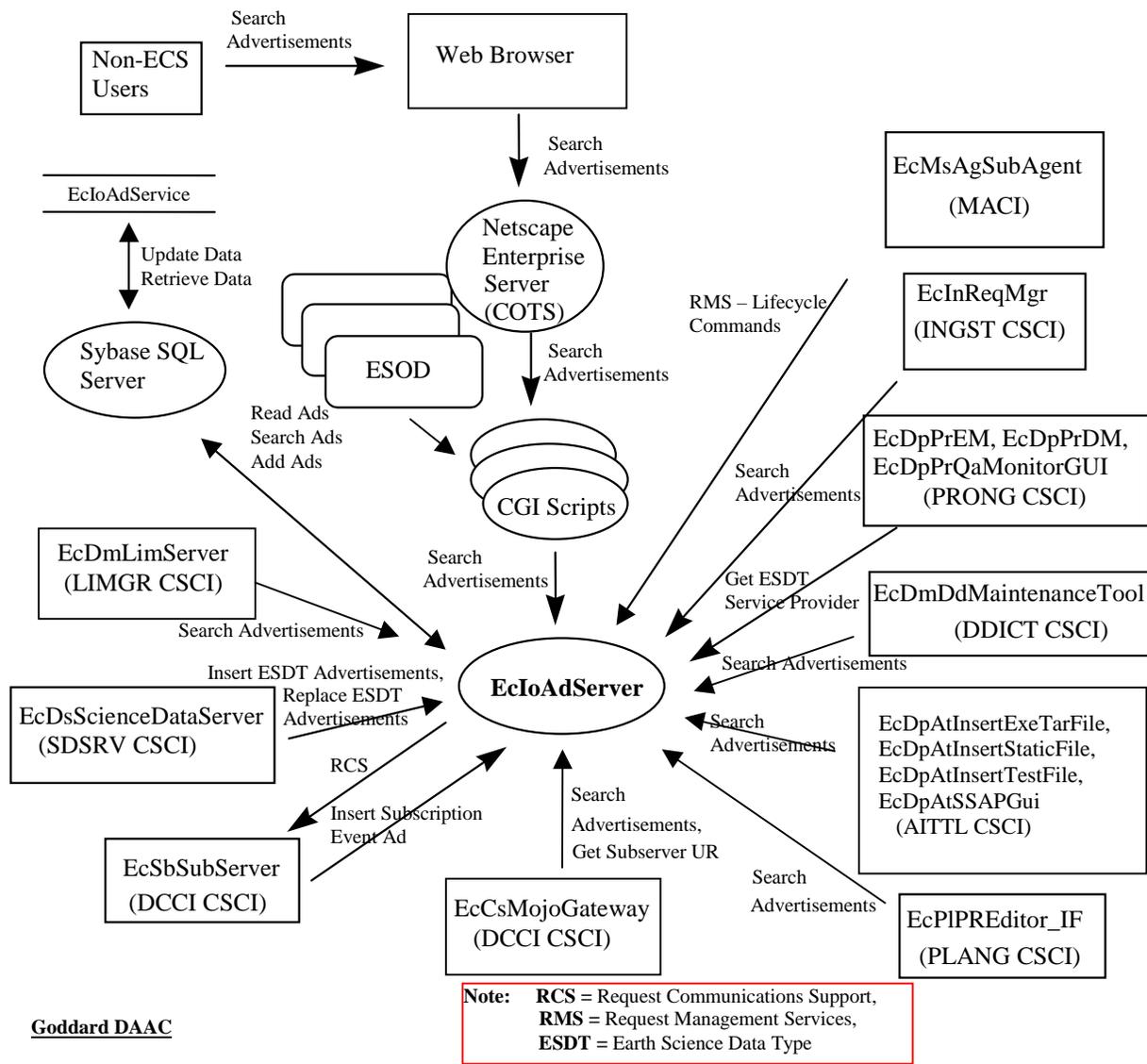


Figure 4.5.1.3-1. Advertising Service Architecture Diagram

4.5.1.4 Advertising Service Process Description

Table 4.5.1.4-1 provides descriptions of the processes shown in the Advertising Service architecture diagram.

Table 4.5.1.4-1. Advertising Service Processes

Process	Type	COTS/ Developed	Functionality
EcloAdServer	Server	Developed	<p>The Advertising Server is the only CSCI within the Interoperability subsystem. Users of the Advertising Server use it for searching, inserting, deleting and updating various types of advertisements. The Advertising server uses a relational DBMS server (Sybase) for persistent storage of the advertisements. The Sybase server is shared with the DMS software configuration items.</p> <p>The Advertising Service offers two basic interfaces</p> <ul style="list-style-type: none"> • Advertising Search: The Advertising Server allows a user to specify search requests on the Advertising database. Searches include searches for data, signature service and subscription event advertisements. • Advertisement Insert and Delete: Provides a user with the capability to insert and delete advertisements within the Advertising database. <p>The Advertising Server supports:</p> <ul style="list-style-type: none"> • Multiple concurrent requests • Synchronous request processing • Asynchronous request processing
Netscape Enterprise Server	Server	COTS	<p>The Netscape Enterprise Server runs at the DAACs and receives and interprets the Hypertext Transport Protocol (HTTP) from the ESOD web pages. Refer to Netscape Server administration documentation for further information.</p>
CGI	CGI	Developed	<p>The Advertising Service user interface uses generic HTML that is accessible via common web browsers (no JAVA involved). The Earth Science On-Line Directory (ESOD) is the CSC that uses the HTML Framework to build the actual HTML files that are viewed by the users using a Web browser. There are a number of CGI programs associated with the HTML interfaces. The ESOD HTML interface communicates with the Advertising Server through the use of CGI programs. The CGI programs are run on the ADSHW CI after being spawned from the Netscape Enterprise Server. A number of these CGIs exist within the Advertising Service for forwarding requests to the Advertising server and receiving results back. The CGI program process names are loAdEsodamGroups, loAdEsodamGroupSearch, loAdEsodamModeration, loAdEsodamModerationForm, loAdEsodamModerationGroups, loAdEsodamModerationQueue, loAdEsodamObsoleteReqs, loAdEsodContributionForm, loAdEsodContributions, loAdEsodEntryDetail, loAdEsodExamples, loAdEsodGroupDetail, loAdEsodScienceSearch, loAdEsodScienceSearchForm, loAdEsodTextSearch, loAdEsodTextSearchForm, loAdEsodUpdateTemplate, loAdInstallForm and loAdEsodWhatsNew.</p>
Sybase Server	Server	COTS	<p>The Sybase Server acts as a SQL server for the Advertising Service. Refer to Sybase documentation for details.</p>

4.5.1.5 Advertising Service Interface Descriptions

Table 4.5.1.5-1 provides descriptions of the interface events shown in the Advertising Service architecture diagram.

Table 4.5.1.5-1. Advertising Service Process Interface Events (1 of 4)

Event	Event Frequency	Interface	Initiated By	Event Description
Search Advertisements	One per request to search advertisements	<i>Process:</i> EcloAdServer <i>Library:</i> IoAdSearch <i>Class:</i> IoAdApprovedAdvSearchcommand <i>CUSTOM libraries:</i> IoAdSearch, IoAdCore, IoAdSubs	PLS Process: EcPIPREditor_IF <i>PLS Library:</i> PICore1 <i>PLS Class:</i> PIDataType DMS Process: EcDmDdMaintenanceTool <i>DMS Library:</i> DmLmReqProc <i>DMS Classes:</i> DmLmProductPlan, DmDdMtDatasetGroup INS Process: EcInReqMgr <i>INS Library:</i> InUpdateUR <i>INS Class:</i> InUpdateUR DPS Processes: EcDpAtInsertTestFile, EcDpAtInsertExeTarFile, EcDpAtInsertStaticFile EcDpAtSSAPGui <i>Libraries:</i> PICore1 PICore1IF <i>Class:</i> DpAtDsrv, PIDataType	The EcloAdServer receives requests to search and retrieve advertisements from the EcPIPREditor_IF, EcInReqMgr, EcDmDdMaintenanceTool, EcDpAtInsertExeTarFile, EcDpAtInsertStaticFile, EcDpAtInsertTestFile, EcDpAtSSAPGui, EcDmLimServer and EcCsMojoGateway.. Non-ECS users also search for advertisements, which are essentially directory searches for the types of data that exist in the system using CGI scripts via the Netscape Enterprise Server.

Table 4.5.1.5-1. Advertising Service Process Interface Events (2 of 4)

Event	Event Frequency	Interface	Initiated By	Event Description
Request Management Services	One per command to start or stop an application	<i>Process:</i> EcloAdServer	<i>Process:</i> EcMsAgSubAgent <i>Library:</i> EcAglstrm <i>Class:</i> EcAgManager <i>Script:</i> EcloAdServer	The EcMsAgSubAgent provides a basic management library of services to the processes, implemented as client or server applications, using the DCCI CSCI Process Framework. The basic management library of services include: <ul style="list-style-type: none"> • Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training).
Get ESDT Service Provider	One per advertising search	<i>Process:</i> EcloAdServer <i>Libraries:</i> IoAdcore, IoAdSubs <i>Classes:</i> IoAdSignatureServiceAdv, IoAdApprovedAdv, IoAdGroup, IoAdProvider	<i>Processes:</i> EcDpPrDM, EcDpPrEM, EcDpPrQaMonitorGUI <i>Library:</i> PICore1 <i>Class:</i> PIDataType	The EcDpPrDM, EcDpPrEM, and EcDpPrQaMonitorGUI send requests to the EcloAdServer, using the Universal Reference obtained from the EcDsScienceDataServer, for a particular ESDT.

Table 4.5.1.5-1. Advertising Service Process Interface Events (3 of 4)

Event	Event Frequency	Interface	Initiated By	Event Description
Get Subserver UR	One per request	<i>Process:</i> EcIoAdServer <i>Library:</i> IoAdvSearch <i>Class:</i> EcIoAdSearch	<i>Process:</i> EcCsMojoGateway <i>Library:</i> EcCsMojoGateway <i>Class:</i> EcMjEcsAdsrvProxy	The EcCsMojoGateway submits a request to retrieve the correct subscription server UR from the EcIoAdServer.
Insert Subscription Event Ad	One per request to insert advertisement	<i>Process:</i> EcIoAdServer <i>Libraries:</i> IoAdCore, IoAdSubs <i>Classes:</i> IoAdSignatureServiceAdv, IoAdApprovedAdv, IoAdGroup, IoAdProvider	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbSr <i>Class:</i> EcSbEvent	The EcIoAdServer receives requests to insert subscription event advertisements from the EcSbSubServer. All inserts are performed at the master site Advertising Server only.
Request Communications Support	One per service request.	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbSr <i>Class:</i> EcSbEvent	<i>Process:</i> EcIoAdServer <i>Libraries:</i> IoAdCore, IoAdSubs <i>Classes:</i> IoAdSignatureServiceAdv, IoAdApprovedAdv, IoAdGroup, IoAdProvider	The DCCI CSCI Process Framework provides a library of services available to each SDPS and CSMS process. The services required to perform the specific process assignments are requested by the process from the Process Framework. These services include: DCE support, file transfer services, Network & Distributed File Services, Bulk Data transfer services, file copying services, name/address services, password services, Server Request Framework (SRF), UR, Error/Event logging, message passing, Fault Handling services, User Authentication services, Mode information, and retrieving the requested configuration attribute-value pairs from the Configuration Registry for ECS applications that request them.

Table 4.5.1.5-1. Advertising Service Process Interface Events (4 of 4)

Event	Event Frequency	Interface	Initiated By	Event Description
Insert ESDT Advertisements	One per data type being inserted	<i>Process:</i> EcloAdServer <i>Libraries:</i> IoAdcore, IoAdSubs <i>Classes:</i> IoAdSignatureServiceAdv, IoAdApprovedAdv, IoAdGroup, IoAdProvider	<i>Process:</i> EcDsScienceDataServer <i>Class:</i> DsDelOSController	The EcloAdServer receives requests to insert ESDT advertisements from the EcDsScienceDataServer including both data product and signature service advertisements. All inserts are performed at the master site Advertising Server only.
Replace ESDT Advertisements	One per advertisement update request	<i>Process:</i> EcloAdServer <i>Libraries:</i> IoAdCore, IoAdSubs <i>Classes:</i> IoAdApprovedAdv, IoAdGroup, IoAdProvider, IoAdProduct	<i>Process:</i> EcDsScienceDataServer <i>Class:</i> DsDelOSController	The EcloAdServer receives requests to update advertisements for data types (ESDTs) from the EcDsScienceDataServer Including both data product and signature service advertisements.
Search Ads/ Read Ads/ Add Ads	One per request to insert advertisement	Sybase Server (COTS)	<i>Process:</i> EcloAdServer COTS libraries: RWDBTools.h CUSTOM libraries: IoAdSearch, IoAdCore, IoAdSubs	The EcloAdServer sends requests to search, insert, update and retrieve advertisements from the Sybase Server.
Update Data	One per update request	COTS: Standard SQL Engine	Sybase SQL Server	The Sybase Server updates data persistently stored on disk based on update requests from the Advertising Server.
Retrieve Data	One per search query	COTS: Standard SQL Engine	Sybase SQL Server	The Sybase Server retrieves data persistently stored on disk based on search queries from the Advertising Server.

4.5.1.6 Advertising Service Data Stores

Table 4.5.1.6-1 provides descriptions of the data stores shown in the Advertising Service architecture diagram.

Table 4.5.1.6-1. Advertising Service Data Stores

Data Store	Type	Functionality
EcloAdService	Database	<p>The Advertising Service database, EcloAdService, is a Sybase relational database that persistently stores the advertisements and advertisement related information on a physical disk.</p> <p>The types of data stored in the Advertising Service database include:</p> <ul style="list-style-type: none">• Data: A list of all the data collections along with their associated metadata within the ECS.• Signature Services: Signature services include the signature required for one server to obtain the services of another server. One example is the acquire signature required for users of the DSS' Science Data Server (EcDsScienceDataServer) to obtain data granules.• Subscription Events: Users or servers within the ECS can subscribe to and be notified of available data.

4.5.2 Interoperability Subsystem Hardware Components

4.5.2.1 Interface Hardware CI (INTHW) Description, as used by the Interoperability Subsystem

The INTHW CI consists of two Interface Servers. In addition, the Interface Servers support the Client Subsystem and a portion of the Communication Subsystem software components. Client and Communication Subsystem related topics are discussed in their respective sections.

The Interface Servers are SUN Server class machines. Detail specifications can be found per the site-specific hardware design diagram, base-line document number 920-TDx-001. Because of their common configuration, these hosts can be configured interchangeably. The ADSRV is the only Interoperability software component that runs on these systems. The Advertising Service provides management of Earth Science related advertisements.

Detailed mappings can be found per the site-specific hardware/software mapping, base line document number 920-TDx-002.

A SUN SPARC Storage Array is dual ported between both hosts and provides storage for the Advertising Database and Sybase Replication components. A detailed configuration is specified per disk partition, base-line document number 922-TDx-009.

In general, custom code and applications are loaded on the internal disks of all hosts. This prevents dependencies on specific hosts or any peripherals.

4.6 Planning Subsystem Overview

The Planning Subsystem (PLS) manages the data production activities at ECS sites in support of the operations staff by providing the following capabilities:

- Identifies the data processing tasks (via data processing requests) performed by a site
- Generates the data production plans for scheduling the identified processing tasks according to different production rules, which define how a particular Product Generation Executive (PGE) is to be run
- Coordinates data production with the DSS and the DPS to achieve an automated production system.

Planning Subsystem Context Diagram

Figure 4.6-1 is the context diagram for the PLS. The diagram shows the events sent to other SDPS and CSMS subsystems and the events the PLS receives from other SDPS and CSMS subsystems. Table 4.6-1 provides descriptions of the interface events shown in the Planning Subsystem Context Diagram.

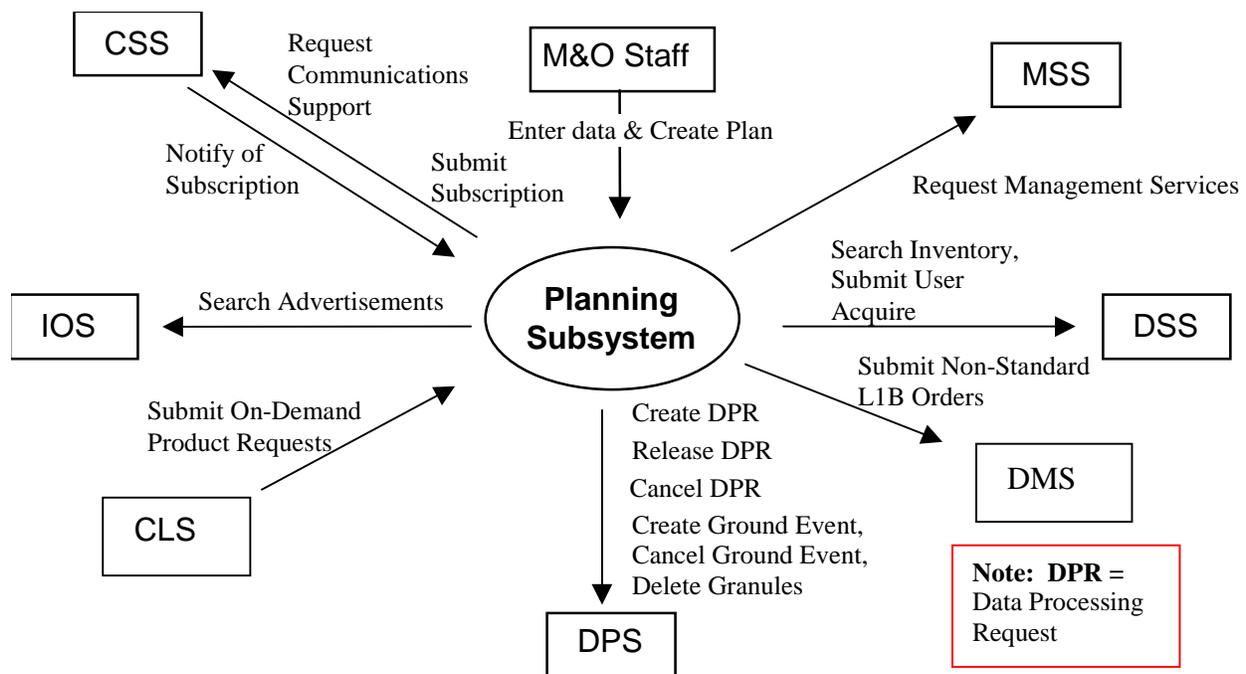


Figure 4.6-1. Planning Subsystem Context Diagram

Table 4.6-1. Planning Subsystem Interface Events (1 of 2)

Event	Interface Event Description
Enter data & Create Plan	The M&O staff enter production request data and issues commands to control the creation of a Production Plan.
Request Management Services	<p>The MSS provides a basic management library of services to the subsystems, implemented as client or server applications, using the CSS Process Framework. The basic management library of services include:</p> <ul style="list-style-type: none"> • Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training).
Search Inventory	The PLS sends inventory search or inspect requests to the DSS to search the ECS inventory/archives (granules). In response, the PLS receives URs for the respective granules satisfying the search.
Submit User Acquire	The PLS submits an acquire command to the DSS on behalf of the user. The user gets a response via the DSS upon data distribution.
Submit Non-Standard L1B Orders	The PLS submits requests through the DMS for the production of non-standard L1B On-Demand Products.
Create DPR	The PLS sends, to DPS, the Data Processing Request Identification (dprId) and whether the DPR is waiting for external input data.
Release DPR	The PLS sends the dprId to the DPS for DPR release.
Cancel DPR	The PLS sends a request to cancel the dprId to the DPS for the deletion of a DPR.
Create Ground Event	The PLS sends the ground event id, resource id, and start time to the DPS to create a ground event to perform maintenance activities on data processing resources.
Cancel Ground Event	The PLS sends the ground event id, resource id, and start time to the DPS to cancel a ground event.
Delete Granules	The PLS sends requests to the DPS to delete granules associated with cancelled DPRs.
Submit On-Demand Product Requests	The CLS submits the on-demand request to the PLS. As a result, the user receives an Order ID. The user receives a notification when the request is processed.
Search Advertisements	The IOS receives search requests for subscription event and signature service advertisements from the PLS. The PLS enters subscriptions with the Subscription Server within the CSS or obtains the proper signatures for acquiring data granules from the DSS (for the insert and update of metadata within the DSS).
Notify of Subscription	A message passing callback in the PLS subscription manager is called, by the CSS, with the UR of the granule inserted into the Data Server as one of the calling parameters.

Table 4.6-1. Planning Subsystem Interface Events (2 of 2)

Event	Interface Event Description
Request Communications Support	The CSS provides a library of services available to each SDPS and CSMS subsystem. The services required to perform the specific subsystem assignments are requested by the subsystem from the CSS. These services include: DCE support, file transfer services, Network & Distributed File Services, Bulk Data transfer services, file copying services, name/address services, password services, Server Request Framework (SRF), UR, Error/Event logging, message passing, Fault Handling services, User authentication services, Mode information, and retrieving the requested configuration attribute-value pairs from the Configuration Registry for ECS applications that request them.
Submit Subscription	The PLS creates a subscription, sent to the CSS, using the advertisement for subscribing to an insert event for an ESDT. In response, PLS receives a corresponding subscription identifier.

The following paragraphs describe the relationships between the PLS and other SDPS subsystems.

DPS Interface

The PLS uses a database link with the DPS Processing CSCI to describe the Product Generation Executives (PGEs) needed to fulfill the production goals. A Data Processing Request (DPR) describes a PGE run to the DPS. A DPR describes the specific input granules, output filenames, and run-time parameters for a PGE, as well as dependencies and predicted run-times. The DPS provides status and processing completion information to the PLS.

DSS Interface

The PLS queries the DSS inventory for data required for processing. If the data exists, the DSS responds to the PLS with granule information (identification, metadata, and location). If the data does not exist, an error message or notification is sent to the PLS.

CSS Interface

The CSS Subscription server provides a notification on the arrival of ECS data. The ECS Advertising service provides the advertisement data required by the PLS to generate subscriptions. The PLS exchanges mode management information with and receives event notifications from the CSS.

MSS Interface

The PLS sends fault management, accounting, security, and performance data to the MSS for logging. The PLS receives Order tracking information for On-Demand Products.

CLS Interface

The PLS receives requests for On-Demand Products from the CLS (ODFRM) and an Order Id is returned to the user.

Planning Subsystem Structure

The PLS is comprised of one CSCI, Production Planning (PLANG CSCI) and one hardware CI, Production Planning (PLNHW).

The Planning and Data Processing Subsystems (PDPS) database resides in the PLNHW and serves both planning and scheduling activities.

Use of COTS in the Planning Subsystem

- Hughes- Delphi Scheduling Class Libraries.

The PLS uses Delphi for scheduling of the Resource Planning Workbench and the Production Planning Workbench. Delphi uses C++ classes to provide user-oriented, integrated, and modular planning and scheduling software utilities.

- RogueWave's Tools.h++

The Tools.h++ class libraries provide libraries of object strings and collections. These libraries must be installed for the PLS processes to run.

- RogueWave's DBTools.h++

The DBTools.h++ C++ class libraries interact with the Sybase database Structured Query Language (SQL) server and buffer the processes from the relational database used.

- ICS' Builder Xcessory

The Builder Xcessory GUI builder tool modifies displays. The Builder generates the C++ code to produce the Mtool display at run time. There is no operational component of Builder Xcessory needed at run-time.

- Sybase Server

The Sybase SQL server provides the capabilities to insert, update and delete PDPS database content. The Sybase SQL Server must be operational during the PLS operations.

- DCE Client

DCE Client provides PLS with communications between other subsystems. DCE can reside on one or both sides of the interface. An instance must be installed on the platform where PLS resides. Although the DCE Client is part of CSS, this COTS item must be installed for PLS to run in the SDPS operational and test environment.

4.6.1 Production Planning (PLANG) Software Description

4.6.1.1 Production Planning Functional Overview

The PLANG CSCI manages the data production activities at each site by providing the Maintenance and Operations (M&O) staff with the following capabilities:

- Defining the data processing tasks (via data processing requests) to perform at the site
- Generating data production plans for scheduling processing tasks

- Coordinating data production with the DSS and the DPS to automate the production system.

The On-Demand Manager is used to manage the on-demand orders received from the ODFRM. Upon receipt of an order, the On-Demand Manager determines the order type. The order type can be a Non-Standard L1B, a DEM, or a higher-level product order. Once the order type is determined, the On-Demand Production Request class verifies that the inputs provided by the ODFRM are valid. The ODPRM periodically checks each order it is tracking to determine if all of the required inputs are available.

If the order is a Non-standard L1B, the ODPRM creates a dummy production request and DPR along with a placeholder granule. When the non-standard L1B product is inserted, the SubMgr updates the placeholder granule with an UR and marks it as accepted.

If the order is a DEM, the ODPRM simply creates the associated dummy DPRs and granules and returns a MSS order id. Once the request is completed the operator inserts the DEM into the SDSRV and invokes the command line tool again to update the user

If the order is for a higher level product, the ODPRM creates the PRs and DPRs necessary to fulfill the order. If all data is available, the production request(s) are added to the list of On-Demand requests maintained by the manager, the data processing request(s) are submitted to the Data Processing Subsystem and the product is produced. As each DPR changes state, the MSS HP OpenView COTS product is updated to reflect the new state and the overall order status is updated accordingly.

4.6.1.2 Production Planning Context

Figure 4.6.1.2-1 is the PLANG CSCI context diagram. The diagram shows the events sent to the PLANG CSCI and the events the PLANG CSCI sends to other CSCIs and the M&O staff. Table 4.6.1.2-1 provides descriptions of the interface events shown in the PLANG CSCI context diagram.

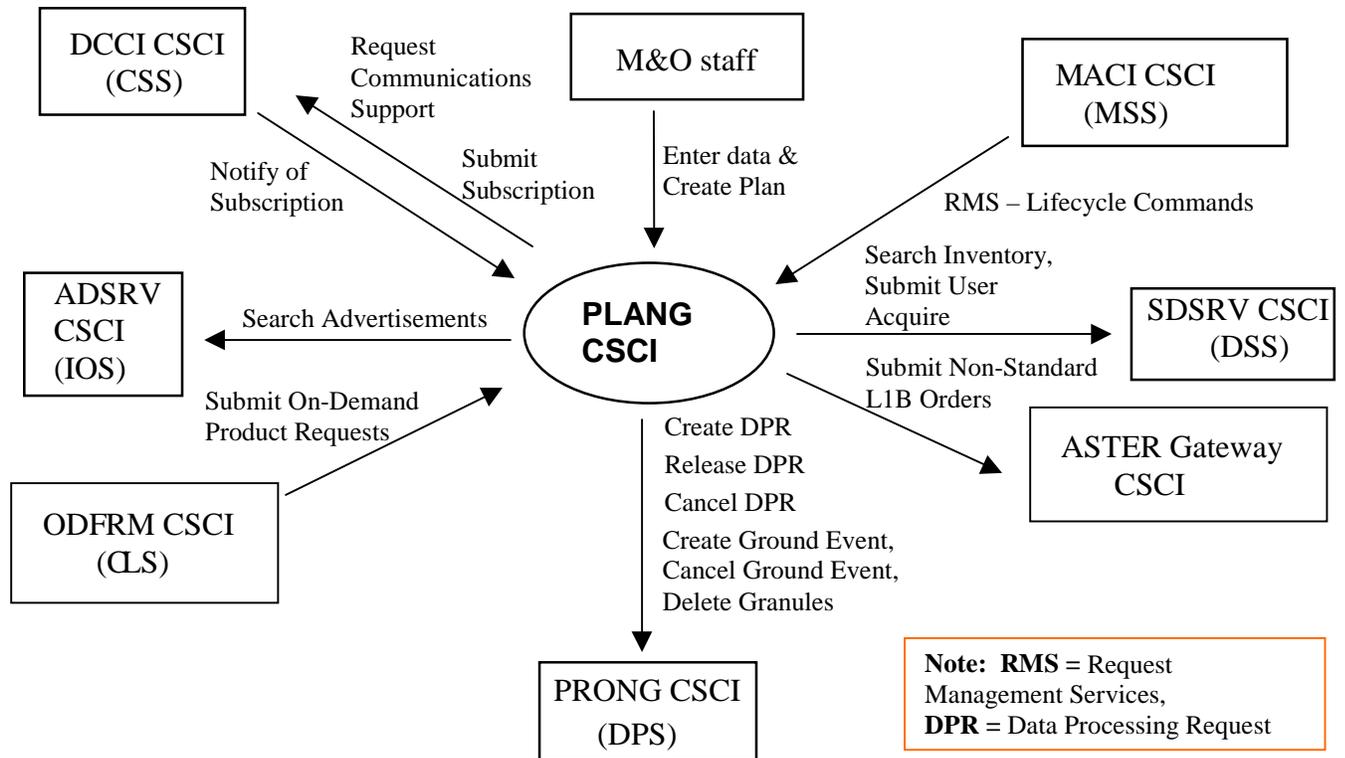


Figure 4.6.1.2-1. PLANG CSCI Context Diagram

Table 4.6.1.2-1. PLANG CSCI Interface Events (1 of 2)

Event	Interface Event Description
Enter data & Create Plan	The M&O staff enter production request data and issues commands to control the creation of a Production Plan.
Request Management Services	The MACI provides a basic management library of services to the CSCIs, implemented as client or server applications, using the DCCI CSCI Process Framework. The basic management library of services include: <ul style="list-style-type: none"> Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training).
Search Inventory	The PLANG CSCI sends inventory search or inspect requests to the SDSRV CSCI to search the ECS inventory/archives (granules). In response, the PLANG CSCI receives granule URs satisfying the search.

Table 4.6.1.2-1. PLANG CSCI Interface Events (2 of 2)

Event	Interface Event Description
Submit User Acquire	The PLANG CSCI submits an acquire command to the SDSRV CSCI on behalf of the user. The user gets a response via the DDIST CSCI upon data distribution.
Submit Non-Standard L1B Orders	The PLANG CSCI submits requests through the ASTER Gateway CSCI for the production of non-standard L1B On-Demand Products.
Create DPR	The PLANG CSCI sends the Data Processing Request Identification (dprId) and whether the DPR is waiting for external input data to the PRONG CSCI.
Release DPR	The PLANG CSCI sends the dprId to the PRONG CSCI.
Cancel DPR	The PLANG CSCI sends a request to cancel the dprId to the PRONG CSCI for the deletion of a DPR.
Create Ground Event	The PLANG CSCI sends the ground event id, resource id, and start time to the PRONG CSCI to create a ground event to perform maintenance activities on data processing resources.
Cancel Ground Event	The PLANG CSCI sends the ground event id, resource id, and start time to the PRONG CSCI to delete a ground event.
Delete Granules	The PLANG CSCI sends requests to the PRONG CSCI to delete granules associated with cancelled DPRs.
Submit On-Demand Product Requests	The ODFRM CSCI submits the on-demand request to the PLANG CSCI. As a result, the user receives an Order ID. The user receives a notification when the request is processed.
Search Advertisements	The ADSRV CSCI receives search requests for subscription event and signature service advertisements from the PLANG CSCI. The PLANG CSCI enters subscriptions with the Subscription Server within the CSS or obtains the proper signatures for acquiring data granules from the SDSRV CSCI (for the insert and update of metadata within the SDSRV inventory).
Notify of Subscription	A message passing callback in the PLANG CSCI subscription manager is called with the granule UR inserted into the SDSRV inventory as a calling parameter.
Request Communications Support	The DCCI CSCI provides a library of services available to each SDPS and CSMS CSCI. The services required to perform the specific CSCI assignments are requested by the CSCI from the DCCI CSCI. These services include: DCE support, File Transfer Services, Network & Distributed File Services, Bulk Data transfer services, File copying services, name/address services, Password services, Server Request Framework (SRF), UR, Error/Event logging, message passing, Fault Handling services, User authentication services, Mode information, and retrieving the requested configuration attribute-value pairs from the Configuration Registry for ECS applications that request them.
Submit Subscription	The PLANG CSCI creates a subscription using the advertisement for subscribing to an ESDT insert event. In response, the PLANG CSCI receives a subscription identifier.

PLANG CSCI interfaces include:

PDPS Database Interface (Common database pseudo-interface with DPS)

The PLS retrieves PGE data stored by the DPS Algorithm Integration and Test Tools (AITTL) CSCI. This PGE data includes the PGE executable, the input data type(s) it requires, the output data type(s) it generates, and the resource requirements (e.g., hardware platform, memory, and disk storage). The PLS uses the PGE data to schedule data processing requests with the DPS.

The PLS deletes DPRs from the PDPS database and some of its associated granules that are not used by other DPRs.

Operator Interface

The Maintenance and Operations (M&O) staff personnel enter Production Requests into the PLS via the Planning User Interface. Production Requests describe the order for data to be produced by the DPS. Production Requests are used to process new data (Routine Production Requests, also known as standing orders) or for reprocessing data (Reprocessing Production Requests). The PLS uses the PGE profile information from the Production Requests to generate the DPRs needed to fulfill the request for data. The Planning User Interface also issues commands to initiate plan creation, plan activation and plan cancellations, and provide reports and status of plan progress. The M&O staff performs resource planning for the entire DAAC through the Planning User Interface with awareness of the impact of ground events on data processing resources.

4.6.1.3 Production Planning Architecture

Figure 4.6.1.3-1 is the PLANG CSCI architecture without the On-Demand Manager included. The diagram shows the events sent to the PLANG CSCI processes and the events sent by the PLANG CSCI processes to other processes. Figure 4.6.1.3-2 is the PLANG CSCI architecture with the On-Demand Manager featured. The diagram shows the events sent to the PLANG CSCI processes and the events sent by the PLANG CSCI processes to other processes.

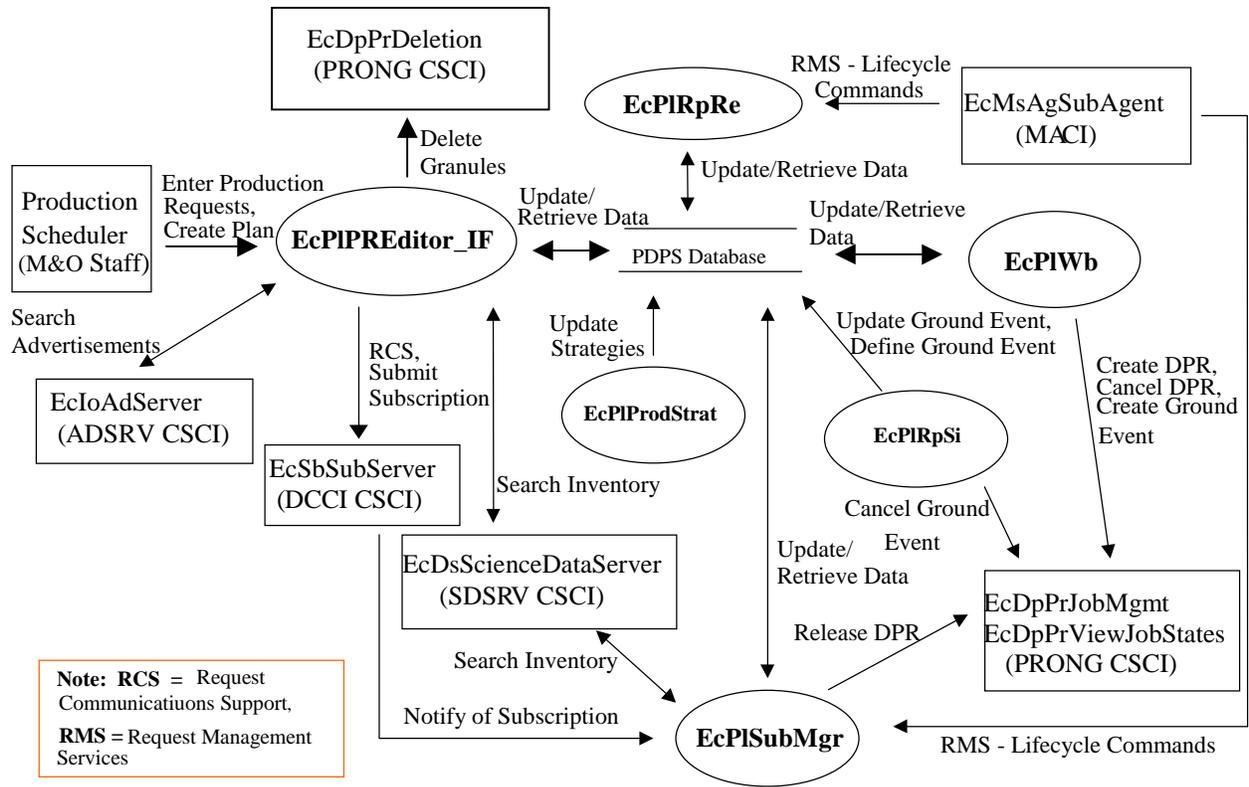


Figure 4.6.1.3-1. PLANG CSCI Architecture Diagram

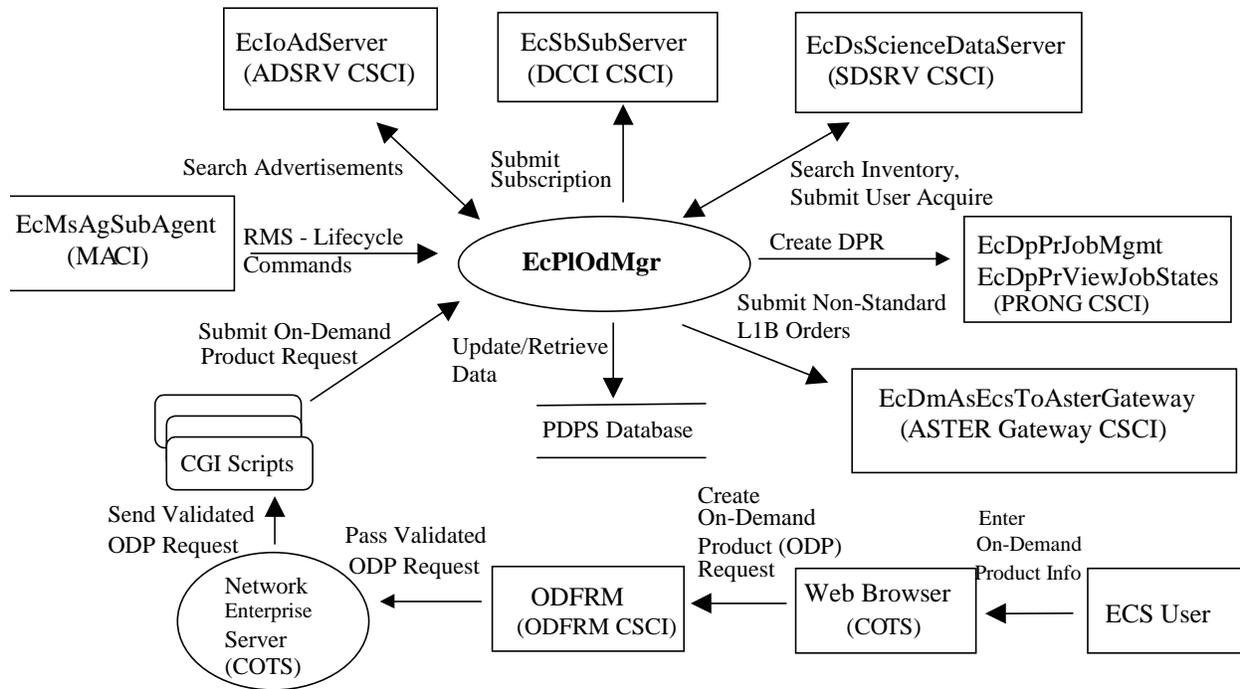


Figure 4.6.1.3-2. PLANG CSCI Architecture Diagram

4.6.1.4 Production Planning Process Descriptions

Table 4.6.1.4-1 provides descriptions of the Production Planning processes shown in the PLANG CSCI architecture diagrams.

Table 4.6.1.4-1. PLANG CSCI Processes (1 of 2)

Process	Type	COTS / Developed	Functionality
EcPIRpRe	GUI	Developed code using Delphi Class Libraries.	<p>The Resource Planning Workbench prepares a schedule for the resources at each respective site, and forecasts the start and completion times of the ground events and the impact on the resources used within the schedule.</p> <p>The workbench allows the M&O staff to:</p> <ul style="list-style-type: none"> • Edit the resources currently available at a site • Associate the resources with production strings (logical groupings of resources used by AutoSys and Data Processing) when allocating resources for a particular PGE.
EcPIWb	GUI	Developed code using Delphi Class Libraries.	<p>The Production Planning Workbench prepares a schedule for the production at a site, and forecasts the start and completion times of the activities within the schedule.</p> <p>Specifically, the Workbench allows:</p> <ul style="list-style-type: none"> • Candidate Plan Creation—from the production requests prepared by the Production Request Editor • Plan Activation—activating a candidate plan • Update of the Active Plan—feedback from the DPS activities are incorporated into the active plan • Cancellation/Modification of the Active Plan. <p>Activating a plan entails rolling a portion of a selected plan into the AutoSys COTS via the DPS. The “schedule” is managed within the DPS. The forecast times generated by the planner are used to set up operator alerts to gross departures from the predicted schedule. Ground Events are sent to the DPS via the EcPIWb.</p>
EcPIPReEditor_IF	GUI	Developed	<p>The Planning User Interface (Production Request Editor) allows the M&O staff to submit production requests to describing the data products to generate. The production request uses the PGE descriptions (profiles) entered during Algorithm Integration and Test (AI&T) to define the Data Processing Requests. The request adds, modifies, and deletes Production Requests, and reviews and modifies the resulting Data Processing Requests. The user specifies rules for producing the individual DPRs for the reprocessing requests. The production request editor is a distinct application and separate from the workbench because defining a production request is unrelated to the planning of a production request.</p>
EcPIRpSi	GUI	Developed code using Delphi Class Libraries	<p>The planning resource gui allows the M&O staff to:</p> <ul style="list-style-type: none"> • Define or cancel ground events (maintenance, etc.) on the allocated resources

Table 4.6.1.4-1. PLANG CSCI Processes (2 of 2)

Process	Type	COTS / Developed	Functionality
EcPISubMgr	Server	Developed	The Subscription Manager receives subscription notifications from the EcDsScienceDataServer via the EcSbSubServer. Subscription notification notifies planning of the arrival of required input data. The Subscription Notification is handled through the Infrastructure message passing service and contains URs pointing to the data objects stored in the EcDsScienceDataServer. The Subscription Manager updates the PDPS database when data is available. When all input data for a DPR is available, the job defined for that DPR is released within the DPS.
EcPIProdStrat	GUI	Developed	The Production Strategies GUI is used to create a set of planning priorities to be applied to each DPR in a plan. This strategy takes user, PGE type, PGE instance, and Production Request priorities into account. This strategy is then saved to the PDPS database.
EcPIOdMgr	Server	Developed	The On-Demand manager receives requests for data from the scientist via the ODFRM web page. The scientist can request a DEM, non-standard L1B or a higher-level product. Once the ODPRM has generated the Production Request (PR) necessary to fulfil the request, in the case of the higher-level product, it submits the PR to DPS for processing. Once DPS is finished, the ODPRM distributes the data to the scientist who requested it.

4.6.1.5 Production Planning Process Interface Descriptions

Table 4.6.1.5-1 provides descriptions of the interface events shown in the PLANG CSCI architecture diagram without the On-Demand Manager included. Table 4.6.1.5-2 provides descriptions of the interface events shown in the PLANG CSCI architecture diagram with the On-Demand Manager featured.

Table 4.6.1.5-1. PLANG CSCI Process Interface Events (1 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
Enter Production Requests	One per production request	<i>Process:</i> EcPIPREditor <i>Library:</i> DpPrDssIF <i>Class:</i> DpPrDSSInterface	M&O staff	The M&O staff request production by selecting a PGE type and the time duration for the PGE to process the input data.

Table 4.6.1.5-1. PLANG CSCI Process Interface Events (2 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
Create Plan	One per created and activated plan	<i>Process:</i> EcPIWb <i>Library:</i> PIWb <i>Class:</i> PIWbScheduler	M&O staff	The M&O staff create and activate a data production plan.
Delete Granules		<i>Process:</i> EcDpPrDeletion <i>Class:</i> DpDeletion	<i>Process:</i> EcPIPReEditor_IF <i>Library:</i> PICore1 <i>Classes:</i> PIWbScheduler, PIDpr	The EcPIPReEditor_IF sends requests to the EcDpPrDeletion process to delete granules associated with a DPR, which has been cancelled.
Request Management Services	One per command to start or stop network applications	<i>Processes:</i> EcPIRpRe, EcPISubMgr	<i>Process:</i> EcMsAgSubAgent <i>Library:</i> EcAgInstrm <i>Class:</i> EcAgManager	<p>The EcMsAgSubAgent provides a basic management library of services to the processes, implemented as client or server applications, using the DCCI CSCI Process Framework. The basic management library of services include:</p> <ul style="list-style-type: none"> • Lifecycle Commands – The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training).

Table 4.6.1.5-1. PLANG CSCI Process Interface Events (3 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
Update/Retrieve Data	One per update/retrieve request	PDPS Database	<p>Process: EcPIRpRe</p> <p>Library: PIRpRe</p> <p>Classes: PIRpReAutosysWin, PIRpReComputerWin, PIRpReDiskWin, PIRpReHardwareWin, PIRpReRealComputerWin, PIRpReStringWin, PIRpReWin</p> <p>Process: EcPIWb</p> <p>Library: PIWb</p> <p>Class: PIWbScheduler</p> <p>Process: EcPISubMgr</p> <p>Library: PICore1</p> <p>Classes: PIDataGranule, PIProductionRequest</p> <p>Process: EcPIPREditor_IF</p> <p>Library: PICore1</p> <p>Class: PIDataType</p>	The EcPIRpRe, EcPIWb, EcPISubMgr, and EcPIPREditor_IF processes send requests to the PDPS database to update/retrieve data defining a PGE. Also, the requests contain information to allow the PGE to be scheduled and executed. Requests are also sent for updates of granule information (location, size, etc.), processing status, and checkpointing.
Update Strategies	One per strategy created.	Active strategies screen	<p>M&O staff</p> <p>Process: EcPIProdStrat</p> <p>Library: PIGUI</p> <p>Class: PIProdStratActive</p>	The M&O staff create strategies when certain jobs need to be prioritized over others. The strategy is saved by name and later can be read by the EcPIWb to prioritize the DPRs in a plan.

Table 4.6.1.5-1. PLANG CSCI Process Interface Events (4 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
Search Advertisements	One per advertisement search request	<i>Process:</i> EcIoAdServer <i>Library:</i> IoAdSearch <i>Class:</i> IoAdApprovedAdvSearchCommand	<i>Process:</i> EcPIPREditor <i>Library:</i> PICore1 <i>Class:</i> PIDataType	The EcIoAdServer process receives requests to search for subscription event and signature service advertisements from the EcPIPREditor_IF process. The EcPIPREditor_IF process obtains the proper signatures for acquiring data granules from the EcDsScienceDataServer for the insert and update of metadata within the DSS inventory.
Submit Subscription	One per subscription created	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbSr <i>Class:</i> EcSbSubscription	<i>Process:</i> EcPIPREditor <i>Library:</i> PICore1 <i>Class:</i> PIDataType	The EcPIPREditor_IF process creates subscriptions using the advertisement for subscribing to an ESDT insert event and enters the subscriptions via the EcSbSubServer.
Notify of Subscription	One per message passing callback	<i>Process:</i> EcPISubMgr <i>Class:</i> PISubMsgCb	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbSr <i>Class:</i> EcSbSubscription	The EcSbSubServer calls a message passing callback in the Subscription Service, with the granule UR inserted into the data server as a calling parameter, to send notification of a subscription event to the EcPISubMgr.

Table 4.6.1.5-1. PLANG CSCI Process Interface Events (5 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
Search Inventory	One per query	<i>Process:</i> EcDsScienceData Server <i>Library:</i> DsCI <i>Class:</i> DsCIQuery	<i>Processes:</i> EcPIPREDitor_IF, EcPISubMgr, EcPIOdMgr <i>Library:</i> DpPrDssIF <i>Class:</i> DpPrDSSInterface	The EcPIPREDitor_IF, EcPISubMgr, and EcPIOdMgr processes create two types of queries. One type only has the ESDT short name and data start and stop times and the other type also includes spatial coordinates. The EcPIPREDitor_IF process queries when the predicted data is available. The EcPIPREDitor_IF process creates an ESDT Reference from an UR after receiving an ESDT Reference from a query. The EcDsScienceDataServer returns ESDT References for granules to satisfy the query. The EcPISubMgr process creates an ESDT Reference from an UR after receiving a subscription notification or receiving an ESDT reference from a query. The EcPISubMgr process queries when predicted data is not available. The EcDsScienceDataServer returns metadata information about the granule being inspected. The EcPIOdMgr process creates an ESDT Reference from an UR after receiving a subscription notification or receiving an ESDT reference from a query. The EcPIOdMgr process queries when predicted data is not available. The EcDsScienceDataServer returns metadata information about the granule being inspected.
Create DPR	One per list of predecessor DPRs	<i>Process:</i> EcDpPrJobMgmt <i>Library:</i> DpPrJM <i>Class:</i> DpPrScheduler	<i>Process:</i> EcPIWb <i>Class:</i> PIWbScheduler	The EcPIWb process sends the dprld and whether the DPR is waiting for external data to the EcDpPrJobMgmt process to create a job in the data production process.
Release DPR	One per dprld send	<i>Process:</i> EcDpPrJobMgmt <i>Library:</i> DpPrJM <i>Class:</i> DpPrScheduler	<i>Process:</i> EcPISubMgr <i>Library:</i> PICore1 <i>Class:</i> PIDpr	The EcPISubMgr process Subscription Manager sends the dprld to the EcDpPrJobMgmt process to start a job in the data production process upon notification of a subscription event occurring.

Table 4.6.1.5-1. PLANG CSCI Process Interface Events (6 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
Cancel DPR	One per dprld send	<i>Process:</i> EcDpPrJobMgmt <i>Library:</i> DpPrJM <i>Class:</i> DpPrScheduler	<i>Process:</i> EcPIWb <i>Library:</i> PICore1 <i>Class:</i> PIDpr	The EcPIWb process sends a request to cancel the dprld to the EcDpPrJobMgmt process for the deletion of a DPR.
Create Ground Event	One per defined ground event	<i>Process:</i> EcDpPrJobMgmt <i>Library:</i> DpPrJM <i>Class:</i> DpPrScheduler	<i>Process:</i> EcPIWb <i>Library:</i> PIWb <i>Class:</i> PIWbScheduler	The EcPIWb process sends the ground event id, resource id, and start time to the EcDpPrJobMgmt process to perform maintenance activities on data processing resources.
Cancel Ground Event	One per defined ground event	<i>Process:</i> EcDpPrJobMgmt <i>Library:</i> DpPrJM <i>Class:</i> DpPrScheduler	<i>Process:</i> EcPIRpSi <i>Library:</i> PIRpSi <i>Classes:</i> PIRpSiScheduler, PIRpSiWinAbs	The EcPIRpSi sends a request to cancel a ground event to the EcDpPrJobMgmt process for the deletion of a ground event.
Define Ground Event	One per request	PDPS Database	<i>Process:</i> EcPIRpSi <i>Library:</i> PIRpSi <i>Classes:</i> PIRpSiScheduler, PIRpSiWinAbs	The EcPIRpSi provides the information to identify the ground events to be done and sends the information to the PDPS database.
Update Ground Event	Once per defined ground event	PDPS Database	<i>Process:</i> EcPIRpSi <i>Library:</i> PIRpSi <i>Classes:</i> PIRpSiScheduler, PIRpSiWinAbs	The EcPIRpSi also updates ground event information in the PDPS database.

Table 4.6.1.5-1. PLANG CSCI Process Interface Events (7 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
Request Communications Support	One per process request.	<i>Process:</i> DCE Security Server <i>Libraries:</i> EcSelogin, EcSeLogincontext <i>Classes:</i> EcSelogin, EcSeLogincontext <i>Library:</i> EcPf <i>Classes:</i> EcPfManagedServer, EcPfclient <i>Library(Common):</i> EcUr <i>Class:</i> EcUrServerUR <i>Process:</i> EcSbSubServer <i>Library:</i> EcSbSr <i>Class:</i> EcSbSubscription <i>Library:</i> EcDcMsgPsng1 <i>Library:</i> Event <i>Class:</i> EcLgErrorMsg <i>Library:</i> EcSeCmi <i>Class:</i> EcSeCmi	<i>Process:</i> EcPIPREditor <i>Library:</i> PICore1 <i>Classes:</i> Most PLS Classes	The Process Framework provides a library of services available to each SDPS and CSMS process. The services required to perform the specific process assignments are requested by the process from the Process Framework. These services include: DCE support, file transfer services, Network & Distributed File Services, Bulk Data transfer services, file copying services, name/address services, password services, Server Request Framework (SRF), UR, Error/Event logging, message passing, Fault Handling services, User authentication services, Mode information, and retrieving the requested configuration attribute-value pairs from the Configuration Registry for ECS applications that request them.

Table 4.6.1.5-2. PLANG CSCI Process Interface Events (1 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
Submit Subscription	One per subscription created	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbSr <i>Class:</i> EcSbSubscription	<i>Processes:</i> EcPIOdMgr <i>Library:</i> PICore1 <i>Class:</i> PIDataType	The EcPIOdMgr processes create subscriptions using the advertisement for subscribing to an ESDT insert event and enter the subscriptions via the EcSbSubServer.
Search Inventory	One per query	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsCI <i>Class:</i> DsCIQuery	<i>Process:</i> EcPIOdMgr <i>Library:</i> DpPrDssIF <i>Class:</i> DpPrDSSInterface	The EcPIOdMgr process creates two types of queries. One type only has the ESDT short name and data start and stop times and the other type also includes spatial coordinates.
Submit User Acquire	One per request	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsCI <i>Classes:</i> DsCIRequest, DsCICommand, DsCIESDTReferenceCollector	<i>Process:</i> EcPIOdMgr <i>Library:</i> PICore2 <i>Classes:</i> PIActivator, DpPrDSSInterface	The EcPIOdMgr submits an acquire command to the EcDsScienceDataServer on behalf of the user. The user gets a response via the EcDsDistributionServer upon data distribution.
Create DPR	One per list of predecessor DPRs	<i>Process:</i> EcDpPrJobMgmt <i>Library:</i> DpPrJM <i>Class:</i> DpPrScheduler	<i>Process:</i> EcPIOdMgr <i>Class:</i> PIActivator	The EcPIOdMgr process sends the dprId and whether the DPR is waiting for external data to the EcDpPrJobMgmt process to create a job in the data production process.
Submit Non-Standard L1B Orders	Per user request	<i>Process:</i> EcDmEcsToAsterGateway <i>Library:</i> DmAsGwEcsReqProc <i>Class:</i> DmGwEcsAsterRequestReceiver DmAsGwEcsProductRequest	<i>Process:</i> EcPIOdMgr <i>Library:</i> PICore2 <i>Class:</i> PiOrderFactory	The EcPIOdMgr submits requests through the EcDmAsEcsToAsterGateway for the production of non-standard L1B On-Demand Products.

Table 4.6.1.5-2. PLANG CSCI Process Interface Events (2 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
Update/Retrieve Data	One per update/retrieve request	PDPS Database	<i>Process:</i> EcPIOdMgr <i>Libraries:</i> PICore1, PIOdMgrClient <i>Class:</i> PIDataType	The EcPIOdMgr process sends requests to the PDPS database to update/retrieve data defining a PGE. Also, the requests contain information to allow the PGE to be scheduled and executed. Requests are also sent for updates of granule information (location, size, etc.), processing status, and check pointing.
Enter On-Demand Product Info	One per user request	<i>Process:</i> Web Interface (COTS)	User	The user fills in the User Information on the Login screen and presses the submit button.
Create On-Demand Product (ODP) Requests	One per user request	<i>Process:</i> ODFRM	<i>Process:</i> Web Browser (COTS)	The ODFRM receives the On-Demand product information and validates the information.
Pass Validated ODP Request	One per user request	<i>Process:</i> Netscape Enterprise Server (COTS)	<i>Process:</i> ODFRM	The Netscape Enterprise Server spawns the process EcCIODRequest with the User Login information.
Send Validated ODP Request	One per user request	CGI Interface, <i>Process:</i> EcCIODRequest	Netscape Enterprise Server (COTS)	The EcCIODRequest process accesses the MSS database and sends the user back the Authentication.
Submit On-demand product request	One per user request	<i>Process:</i> EcPIOdMgr <i>Library:</i> EcPIOnDemandMgr <i>Class:</i> PIODMsgProxy	CGI Interface, <i>Process:</i> EcCIODRequest	The EcCIODRequest process causes an order to be created in the MSS database and sends the request to the EcPIOdMgr, which sends the user back the OrderId.

Table 4.6.1.5-2. PLANG CSCI Process Interface Events (3 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
Request Management Services	One per command to start or stop network applications	<i>Processes:</i> EcPIOdMgr	<i>Process:</i> EcMsAgSubAgent <i>Library:</i> EcAgInstrm <i>Class:</i> EcAgManager	The EcMsAgSubAgent provides a basic management library of services to the processes, implemented as client or server applications, using the DCCI CSCI Process Framework. The basic management library of services include: <ul style="list-style-type: none"> • Lifecycle Commands – The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training).
Search Advertisements	One per advertisement search request	<i>Process:</i> EcIoAdServer <i>Library:</i> IoAdSearch <i>Class:</i> IoAdApprovedAdvSearchCommand	<i>Process:</i> EcPIOdMgr <i>Library:</i> PICore1 <i>Class:</i> PIDataType	The EcIoAdServer process receives requests to search for subscription event and signature service advertisements from the EcPIOdMgr process. The EcPIOdMgr process obtains the proper signatures for acquiring data granules from the EcDsScienceDataServer for the insert and update of metadata within the DSS inventory.

4.6.1.6 Production Planning Data Stores

Table 4.6.1.6-1 provides descriptions of the production planning data stores shown in the PLANG CSCI architecture diagram.

Table 4.6.1.6-1. PLANG CSCI Data Stores

Data Store	Type	Functionality
PDPS Database	Database	<p>The PDPS database is replicated within each site for fault handling purposes. This PDPS database holds all the persistent data (and facilitates the sharing of this data) including, but not limited to:</p> <ul style="list-style-type: none"> • resource information entered with the Resource Planning utilities • PGE and data type information entered at SSIT • Production Request, Data Processing Request and Data Granule information entered using the Production Request Editor • plan information entered using the Production Planning Workbench • task recovery information • Production strategies entered using the Production Strategy GUI <p>The PDPS database also provides security, fault tolerance, and verifies requests for concurrent access to data.</p>

4.6.2 Planning Subsystem Hardware Components

4.6.2.1 Planning Hardware CI (PLNHW) Description

The PLNHW hardware (PLNHW) consists of an SNMP server with the Sybase database management system (DBMS) and the workstations to support the Operations staff by providing the Planning Workbench.

The PDPS DBMS Server runs on either a four processor SUN Server or a dual-processor Sun workstations (see 920-TDx-001 series of baseline documents) with 64-bit Ultra-SPARC processors. Each PDPS DBMS Server is equipped with 512 MB of memory (see 920-TDx-001 series of baseline documents) required by the workstation processors and the Sybase DBMS.

The internal disks provide swap space and file system space for the operating system and the file space for applications software (see 920-TDx-001 series of baseline documents).

Either a Storage Array or an appropriately sized storage unit configured for the database application provides storage for the PDPS database. These storage units are attached to the host via a fast-wide small computer system (SCSI). Additionally, this storage unit backs up the Queuing Server database (see Section 4.7.3.1: Data Processing Hardware).

A Fiber Distributed Data Interface (FDDI) sub-network is implemented at each site to support the PDPS. Each processing unit of SPRHW (including the Queuing Server) is dual-attached to the PDPS FDDI sub-network (see 920-TDx-001, 921-TDx-002, 921-TDx-003, and 921-TDx-004 series of baseline documents).

The dual-ring FDDI implementation provides a fault tolerance capability. Media failures within the FDDI fabric do not result in a loss of service and do not require a re-configuration. With the inherent fault tolerance of FDDI, multiple physical communications paths to each host are not necessary.

4.6.2.2 Planning Workstation Description

The Planning workstation contains and runs the Planning Workbench software. The Planning Workbench is the Production Planning function and the Resource Planning function. One or more workstations are used to run Production Planning and/or Resource Planning at each site based on the site size.

The Planning workstation is a SUN workstation class machine with either a single SPARC or Ultra-SPARC based processor (see 920-TDx-001 series of baseline documents).

The Planning workstation has 384 MB of memory (see 920-TDx-001 series of baseline documents) and each has a fast-wide SCSI controller to attach to a storage subsystem.

The internal disks provide swap space for the operating system and file system space for the operating system and applications (see 920-TDx-001 series of baseline documents). Additional storage for the Planning workstations can be attached to the SCSI controller.

A FDDI sub-network is implemented at each site to support the PDPS. The Planning workstations use a single-attached FDDI interface to connect with the remaining members of the PDPS suite (see 920-TDx-001, 921-TDx-002, 921-TDx-003, and 921-TDx-004 series of baseline documents). The Planning workstations also communicate with other ECS hardware items and the external world via the DAAC FDDI switch.

Sites generally have a minimum of two Planning workstations. If a planning workstation fails, another planning workstation assumes the Planning Workbench functionality from the failed workstation. In cases with one Planning workstation, the Planning workstation tasks are assumed by an available equivalent workstation. Faulty hardware is either repaired or replaced by a certified technician.

4.7 Data Processing Subsystem Overview

The Data Processing Subsystem (DPS) provides the Data Processing capabilities at each ECS site. The DPS capabilities include:

- A queued processing environment to support data product generation. The DPS executes DPRs on available processing resources, as an associated processing job containing all the information needed to accomplish the processing. DPRs are submitted by the PLS and triggered by the arrival of data or triggered internally by the PLS (i.e., reprocessing). PGEs resulting from the integration and test of delivered science algorithms [ref.: ECS White Paper 193-00118] and encapsulated into the SDPS with the Science Data Processing (SDP) Toolkit are used by DPRs to process data. User-specified methods are also used for processing specific data types
- The Operational interfaces required to monitor the execution of the science software (PGEs)
- Support for science algorithm execution via the SDP Toolkit. The SDP Toolkit is a set of tools to provide a common interface for encapsulating each science algorithm into the SDPS environment. (See the [SDP Toolkit Users Guide for the ECS Project \(333-CD-003-002\)](#) and [PGS Toolkit Requirements Specification for the ECS Project \(193-801-SD4-001, a.k.a. GSFC 423-06-02\)](#) for guidance on the roles and responsibilities of the SDP Toolkit to support the execution of science software.
- Support for the preliminary format processing of data sets (L0 data products) required by the science algorithms
- Providing an Algorithm Integration and Test (AI&T) environment to integrate new science algorithms, new versions of existing science algorithms, and user methods into the SDPS environment. The system acquires the algorithm or method via an ingest process reflecting local site policies for acceptance of software for integration into the environment. (See Section 4.7.2 “Algorithm Integration and Test Tools (AITTL) CSCI Description).
- The DAAC Quality Assurance (QA) procedures and conditions to verify each data product by the scientific personnel at each DAAC. All data products, both those generated by and input to a submitted job, are available for examination by DAAC scientific personnel to verify data content to be in accordance with quality standards set by the DAAC.

Data Processing Subsystem Context

Figure 4.7-1 is the Data Processing Subsystem context diagram. The diagram shows the events sent to the DPS and the events the DPS sends to other SDPS and CSMS subsystems and the Operations staff.

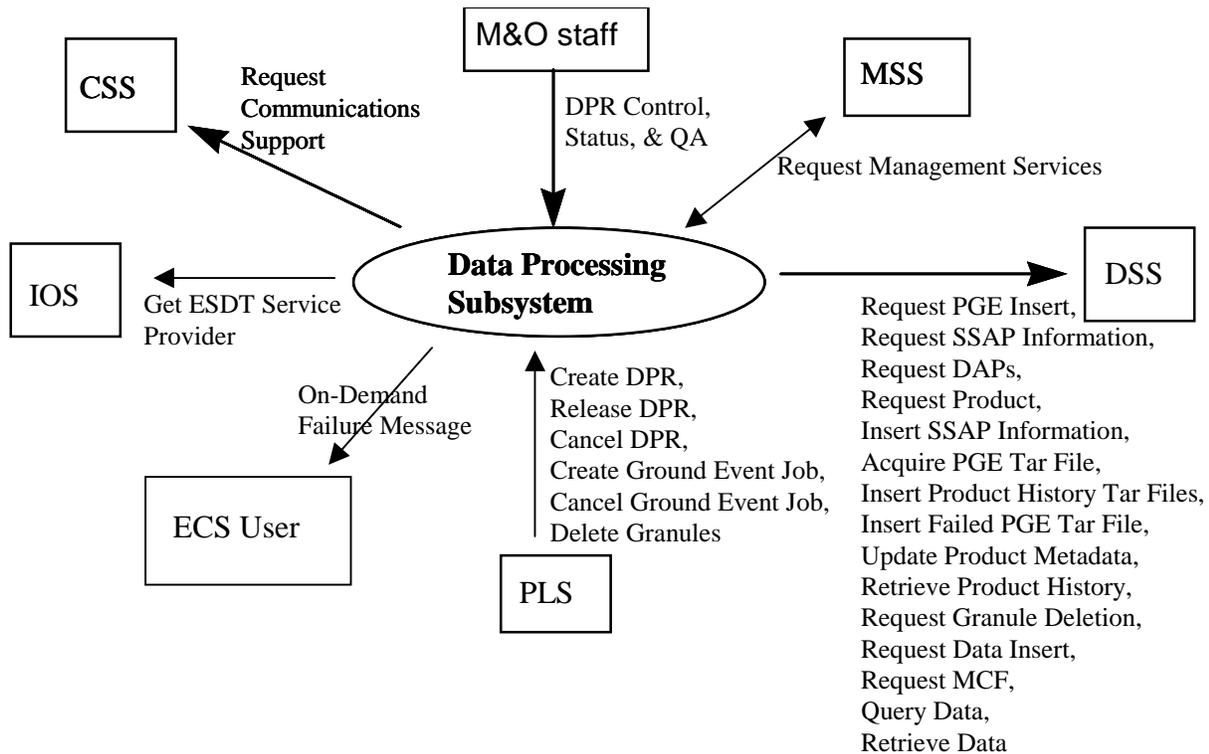


Figure 4.7-1. Data Processing Subsystem Context Diagram

Table 4.7-1 provides descriptions of the interface events shown in the Data Processing Subsystem context diagram.

Table 4.7-1. Data Processing Subsystem Interface Events (1 of 3)

Event	Interface Event Description
DPR Control, Status, & QA	The M&O staff provide Data processing control and supports DPR status and Quality Assurance activities.

Table 4.7-1. Data Processing Subsystem Interface Events (2 of 3)

Event	Interface Event Description
Request Management Services	<p>The MSS provides a basic management library of services to the subsystems, implemented as client or server applications, using the CSS Process Framework. The basic management library of services include:</p> <ul style="list-style-type: none"> • Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training). • Update Request Status - The DPS informs the MSS to update the status of an On-Demand Processing Request, when such request changes status (i.e., from Running to Completed, or from Running to Failure).
Request PGE Insert	The DPS sends requests to the DSS to insert data that defines a PGE and allows it to be scheduled and executed.
Request SSAP Information	The DPS sends requests to the DSS for SSAP information, including names of existing SSAPs and the information associated with a specific SSAP. In response, the DSS sends lists of SSAPs and related information.
Request DAPs	The DPS requests DAPs based on URs from the DSS. The DAPs are placed on a local DPS disk.
Request Product	The DPS sends requests, to the DSS, for particular data granules to be pushed, via the FTP service, onto the DPS science processor as input for data processing or for SSIT work.
Insert SSAP Information	The Operations staff sends requests to the DSS to insert SSAP information, via the DPS SSAP GUI, including SSAP name, SSAP version number, PGE name, PGE version number, and SSAP Acceptance Date.
Acquire PGE tar file	The DPS acquires a tar file for any PGE not currently local to the science processor from the DSS. The executable is extracted from the tar file and used during PGE execution.
Insert Product History Tar Files	The DPS sends a request to the DSS to insert the PGE Production History Tar File resulting outputs for permanent archive after the PGE has successfully completed executing.
Insert Failed PGE Tar File	After an unsuccessful execution of a PGE, the DPS obtains the Tar file containing the PGE log files, core dump (if any), PCF and other files, and requests the files be inserted into the DSS for permanent archive.
Update Product metadata	The Operations Staff uses the QA Monitor GUI in the DPS to send requests to update product metadata in the DSS.
Retrieve Product History	The Operations Staff uses the QA Monitor GUI to submit requests to the DSS to transfer the Production History tar file from the Science Data archives to the user's host machine.
Request Granule Deletion	The DPS sends delete requests, to the DSS, for particular granules (interim data) in the archive and the associated metadata to be deleted from the SDSRV inventory.

Table 4.7-1. Data Processing Subsystem Interface Events (3 of 3)

Event	Interface Event Description
Request Data Insert	The DPS sends requests to the DSS to insert a particular file or files into the archive, and catalog the associated metadata in the SDSRV inventory. These files can be processing output, static files received with PGEs, PGE tar files, APs, SSAPs or DAPs, failed PGE tar files, or production history files.
Request MCF	The INS and DPS request the MCF template, from the DSS, prior to a data insert request.
Query Data	The DPS submits requests of this type to the DSS. It searches the archive for granules that match the user-supplied selection criteria: data type and begin/end date. Results are displayed to the user.
Retrieve Data	The DPS sends retrieval requests, to the DSS, for a particular data granuleId. The product is transferred (pushed), via the File Transfer Protocol (FTP) service, onto the DPS science processor and used as input for Product Generation Executive (PGE) processing or for Science Software Integration and Test (SSIT) work.
Create DPR	The DPS uses the dprld to insert a job box for a DPR into AutoSys if all the required input data is available. If the input data is not available, information used to construct the job in AutoSys is queued by the Job Management CSC until a release DPR is received.
Release DPR	The DPS uses the dprld to release jobs currently waiting for external data in the Job Management queue into AutoSys.
Cancel DPR	The DPS uses the dprld to delete jobs in AutoSys or from the queue.
Create Ground Event Job	The DPS uses the ground event Id to create a ground event job in AutoSys.
Cancel Ground Event Job	The DPS uses the ground event Id to delete a ground event job in AutoSys.
Delete Granules	The PLS sends requests to the DPS to delete granules associated with cancelled DPRs.
On-Demand Failure Message	The DPS informs the end user (the submitter) of an On-Demand Processing Request when such a request has an unrecoverable failure.
Get ESDT Service Provider	The IOS receives requests to search for signature service advertisements from the DPS. The DPS obtains the proper signatures for communicating with the DSS.
Request Communications Support	The CSS provides a library of services available to each SDPS and CSMS subsystem. The services required to perform the specific subsystem assignments are requested by the subsystem from the CSS. These services include: DCE support, file transfer services, Network & Distributed File Services, Bulk Data transfer services, file copying services, name/address services, password services, Server Request Framework (SRF), UR, Error/Event logging, message passing, Fault Handling services, User Authentication services, Mode information, and retrieving the requested configuration attribute-value pairs from the Configuration Registry for ECS applications that request them.

The DPS has an internal interface to the COTS software product AutoSys. The DPS creates, starts, and deletes job boxes in AutoSys via this interface.

The PLS determines the processing activities required to generate data products specified by the Operations staff in a Production Request. Each processing activity is called a DPR. The PLS creates, releases or cancels DPRs in AutoSys via the DPS.

The DSS accesses the data archives via authorized user requests. The DPS requests the required input data for a PGE and Metadata Configuration Files (MCFs) from the DSS. The DPS also inserts PGE generated products, provides product production histories, and provides failed PGE information for debugging purposes. The DPS uses the DSS as a permanent repository for PGE tar files, Algorithm packages (APs), Science Software Archive Packages (SSAPs) and Delivered Algorithm Packages (DAPs).

Data Processing Subsystem Structure

The DPS is comprised of three CSCIs:

- The Processing (PRONG) CSCI manages and monitors the Science Data Processing (SDP) environment to execute Science Software and algorithms (called PGEs) and generates data products.
- The Algorithm Integration and Test Tools (AITTL) CSCI is a set of tools for test and integration of new science software, new versions of existing science software, and user methods in the SDP operational environment. AITTL combines custom developed code with COTS software starting from a central application called the SSIT Manager.
- The SDP Toolkit (SDPTK) CSCI provides a set of software libraries to integrate Science Software into the SDPS environment. By promoting the POSIX standard, these libraries allow the SDP environment to support the generation of data products in a heterogeneous computer hardware environment. (See [SDP Toolkit Design Specification](#) (455-TP-001-001) for the SDPTK architecture).

Use of COTS in the Data Processing Subsystem

- **Platinum Technology's AutoSys** is a job scheduling software application to automate operations in a distributed UNIX environment. AutoSys performs automated job control functions for scheduling, monitoring, and reporting on the jobs residing on any Unix machine attached to an ECS network on the Science Data Processing hardware. AutoSys provides job-scheduling support with an Operator Console for monitoring and human intervention in the job stream. The Operator console allows the M&O staff to restart failed jobs and to view the status of events related to the job's execution. The Operator console includes an alarm manager, set in the job definition, to assist the Operations staff when responding to fault situations.
- **Platinum Technology's AutoXpert** is a GUI providing different methods of viewing a job schedule progress. Noting color changes on the JOBSCAPE GUI can monitor the progression of DPR execution. Failed jobs can be detected and restarted if the job has failed due to the unavailability of an external resource. The HostScope GUI can be used to view the status of the science processors.

- **Sybase Server**

The Sybase SQL server provides the capabilities to insert, update and delete PDPS database content. The Sybase SQL Server must be operational during the DPS operations.

- **DCE Client**

DCE Client provides DSS with communications between other subsystems. DCE can reside on one or both sides of the interface. An instance must be installed on the platform where DSS resides. Although the DCE Client is part of CSS, this COTS item must be installed for DSS to run in the SDPS operational and test environment.

The DPS provides the hardware resources for science software execution, queuing, dispatching, and managing in a distributed environment of computing platforms. The DPS hardware comprises three hardware CIs:

- **Science Processor** - The Science Processor HWCI (SPRHW) contains processing resources (central processing units, memory, disk storage, and input/output subsystems) necessary to perform first-time processing, reprocessing, and Algorithm Integration and Test (AI&T). Also, SPRHW provides the hardware resources (a Queuing Server) to support management of the science processes.
- **Algorithm Quality Assurance** - The Algorithm Quality Assurance HWCI (AQAHW) supports the DAAC Operations staff in performing the planned science and non-science product data quality validation procedures.
- **Algorithm Integration and Test** - The AI&T HWCI (AITHW) resources provide the operating system and support for the integration and test of science software at each DAAC. AITHW is the workstations and hardware tools required for software integration and test. AITHW does not, in this case, provide the computer capacity required for science software test (SPRHW provides the test capacity).

4.7.1 Processing Software Description

4.7.1.1 Processing Functional Overview

The Processing (PRONG) CSCI initiates, monitors, and manages the execution of science software algorithms (referred to as PGEs). The PRONG CSCI is informed of the required execution of a PGE through a DPR received from the PLS. When all necessary input data becomes available, PRONG initiates the execution of the PGE. (N.B.: Some or all input data can reside in a Data Server not at the DAAC site.)

The PRONG CSCI has the following capabilities:

- Manages execution of science software algorithms
- Manages SDP computer hardware resources
- Manages the data flow required to execute a science software algorithm
- Manages the data flow generated by the execution of a science software algorithm

- Monitors processing status, and allows manual intervention, when necessary, in the SDP operations environment, including processing queue control
- Supports validation of product data quality
- Provides status and user updates (in the event of a failure) for On-Demand Processing Requests

4.7.1.2 Processing Context

Figure 4.7.1.2-1 is the PRONG CSCI context diagram. The diagram shows the events sent to the PRONG CSCI and the events the PRONG CSCI sends to other CSCIs and the Maintenance and Operations (M&O) staff.

Table 4.7.1.2-1 provides descriptions of the interface events shown in the PRONG CSCI context diagram.

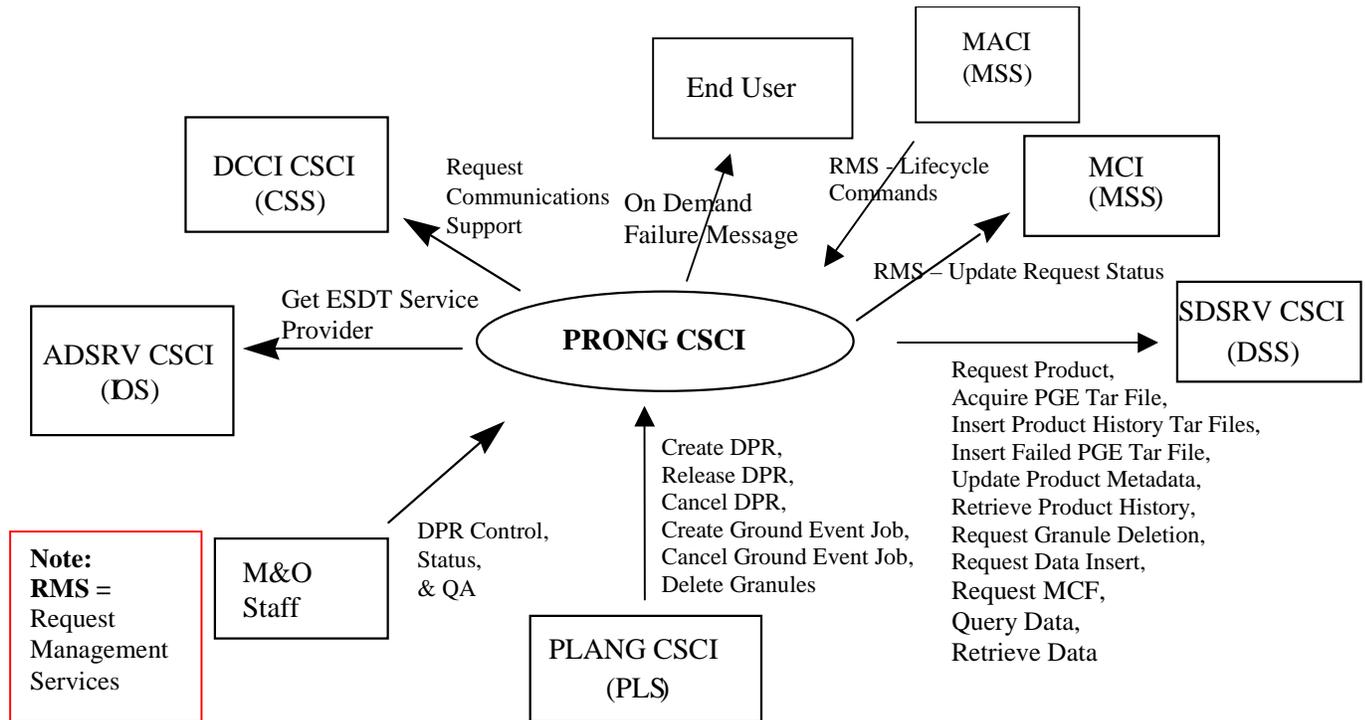


Figure 4.7.1.2-1. PRONG CSCI Context Diagram

Table 4.7.1.2-1. PRONG CSCI Interface Events (1 of 2)

Event	Interface Event Description
On-Demand Failure Message	The PRONG CSCI informs the end user (the initiator of the request) if an On-Demand Processing Request fails, if such failure is unrecoverable.
Request Management Services	<p>The MCI and MACI provide a basic management library of services to the CSCIs, implemented as client or server applications, using the DCCI CSCI Process Framework. The basic management library of services include:</p> <ul style="list-style-type: none"> • Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training). • Update Request Status - The DPS informs the MSS to update the status of an On-Demand Processing Request, when such request changes status (i.e., from Running to Completed, or from Running to Failure).
Request Product	The PRONG CSCI sends requests, to the SDSRV CSCI, for particular data granules to be pushed, via the FTP service, onto the DPS science processor as input for data processing or for SSIT work.
Acquire PGE tar file	The PRONG CSCI acquires a tar file for any PGE not currently local to the science processor from the SDSRV CSCI. The executable is extracted from the tar file and used during PGE execution.
Insert Product History Tar Files	The PRONG CSCI sends a request to the SDSRV CSCI to insert the PGE Production History Tar File resulting outputs for permanent archive after the PGE has successfully completed executing.
Insert Failed PGE Tar File	After an unsuccessful execution of a PGE, the PRONG CSCI obtains the Tar file containing the PGE log files, core dump (if any), PCF and other files, and requests the files be inserted into the SDSRV CSCI for permanent archive.
Update Product metadata	The Operations Staff uses the QA Monitor GUI in the PRONG CSCI to send requests to update product metadata in the SDSRV CSCI.
Retrieve Product History	The Operations Staff uses the QA Monitor GUI to submit requests to the SDSRV CSCI to transfer the Production History tar file from the Science Data archives to the user's host machine.
Request Granule Deletion	The PRONG CSCI sends delete requests to the SDSRV CSCI for particular granules (interim data) in the archive and associated metadata to be deleted from the SDSRV inventory.
Request Data Insert	The PRONG CSCI sends requests to the SDSRV CSCI to insert a particular file or files into the archive, and catalog the associated metadata in the SDSRV inventory. These files can be processing output, static files received with PGEs, PGE tar files, APs, SSAPs or DAPs, failed PGE tar files, or production history files.
Request MCF	The PRONG CSCI requests a Metadata Configuration File (MCF) template for each output data type for specific PGEs from the SDSRV CSCI and the MCF template is populated with metadata from the output granule.
Query Data	The PRONG CSCI submits requests of this type to the SDSRV CSCI. It searches the archive for granules that match the user-supplied selection criteria: data type and begin/end date. Results are displayed to the user.

Table 4.7.1.2-1. PRONG CSCI Interface Events (2 of 2)

Event	Interface Event Description
Retrieve Data	The PRONG CSCI sends retrieval requests, to the SDSRV CSCI, for a particular data granuleId. The product is transferred (pushed), via the File Transfer Protocol (FTP) service, onto the DPS science processor and used as input for Product Generation Executive (PGE) processing or for Science Software Integration and Test (SSIT) work.
Create DPR	The PRONG CSCI uses the dprId to insert a job box for a DPR into AutoSys if all the input data required for the DPR is available. If the input data is not available, information used to construct the job in AutoSys is queued by the Job Management CSC until a release DPR is received.
Release DPR	The PRONG CSCI uses the dprId to release jobs currently waiting for external data in the Job Management queue into AutoSys.
Cancel DPR	The PRONG CSCI uses the dprId to delete jobs in AutoSys or from the queue.
Create Ground Event Job	The PRONG CSCI uses the ground event Id to create a ground event job in AutoSys to perform maintenance activities on data processing resources.
Cancel Ground Event Job	The PRONG CSCI uses the ground event Id to delete a ground event job in AutoSys.
Delete Granules	The PLANG CSCI sends requests to the PRONG CSCI to delete granules associated with cancelled DPRs.
DPR Control, Status, & Q/A	The M&O staff controls Data Processing Request (DPR) activity with the capability to cancel, suspend, resume, and modify a DPR. The M&O staff supports status collecting, PRONG hardware resource monitoring and Quality Assurance validating processes.
Get ESDT Service Provider	The ADSRV CSCI receives search requests for signature service advertisements from the PRONG CSCI. The PRONG CSCI obtains the proper signatures and universal references for communicating with the SDSRV CSCI.
Request Communications Support	The DCCI CSCI provides a library of services available to each SDPS and CSMS CSCI. The services required to perform the specific CSCI assignments are requested by the CSCI from the DCCI CSCI. These services include: DCE support, file transfer services, Network & Distributed File Services, Bulk Data transfer services, file copying services, name/address services, password services, Server Request Framework (SRF), UR, Error/Event logging, message passing, Fault Handling services, User Authentication services, Mode information, and retrieving the requested configuration attribute-value pairs from the Configuration Registry for ECS applications that request them.

4.7.1.3 Processing Architecture

Figures 4.7.1.3-1, 4.7.1.3-2, and 4.7.1.3-3 are the PRONG CSCI architecture diagrams. The diagrams show the events sent to the PRONG CSCI processes and the events the PRONG CSCI processes send to other processes. The PRONG CSCI consists of COTS software and ECS developed processes.

The PRONG CSCI has interfaces with:

- Planning Subsystem – The PLS creates a production plan executed by the PRONG CSCI through the use of DPRs. Each DPR represents one processing job performed by a DPS computer resource. The PRONG CSCI provides DPR status information to the PLS to assist in production management activities.
- Data Server Subsystem – The PRONG CSCI supports SDPS data generation by requesting and receiving data (Data Staging) from a Data Server maintaining raw data and generated products. Also, the PRONG CSCI transfers data (Data de-staging) to a Data Server to archive generated data products.
- SDP Toolkit – The PRONG CSCI provides the location of input data and the location for the generated output data products. While a PGE is executing, the PRONG CSCI monitors the execution and provides current status to the M&O staff. Status includes current processing event history (e.g., data staging, and execution). Process monitoring includes checking resource usage by the PGE. At PGE execution completion, the PRONG CSCI initiates the transfer of the generated data product to the respective Data Server.
- System Management Subsystem – The PRONG CSCI relies on the MSS services for resource management and thus provides system management information including fault, accounting, configuration, security, performance, and accountability to the MSS. It also uses the Request Status Tracking capability to update the status of On-Demand Requests made by users.
- Operations – Supports PGE execution management and monitoring and the generation of SDPS Data Products via a Human Machine Interface (HMI). The HMI supports status information collection for a DPR, controlling DPR executions, and monitoring the status of the DPS hardware resources. The HMI also supports manual quality assurance activities performed at the DAAC.

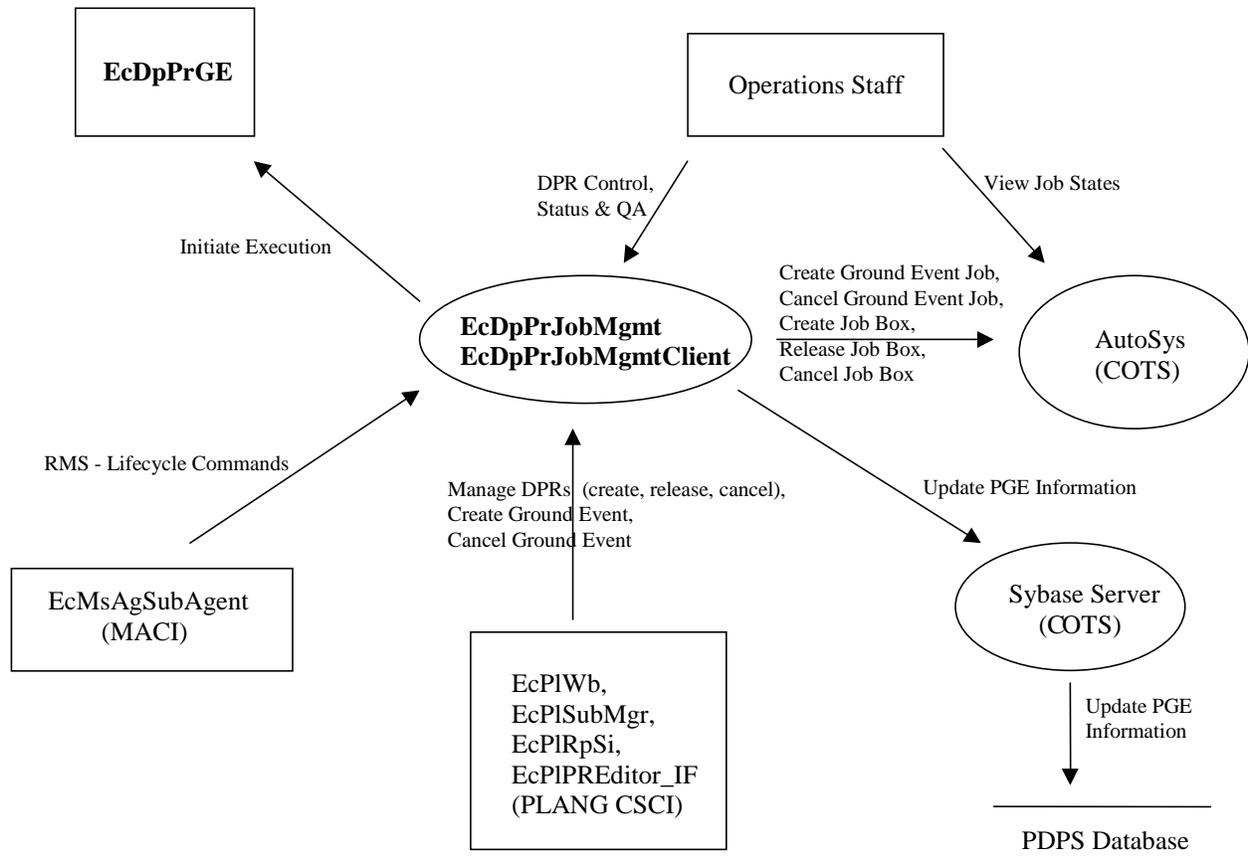


Figure 4.7.1.3-1. PRONG CSCI Architecture Diagram

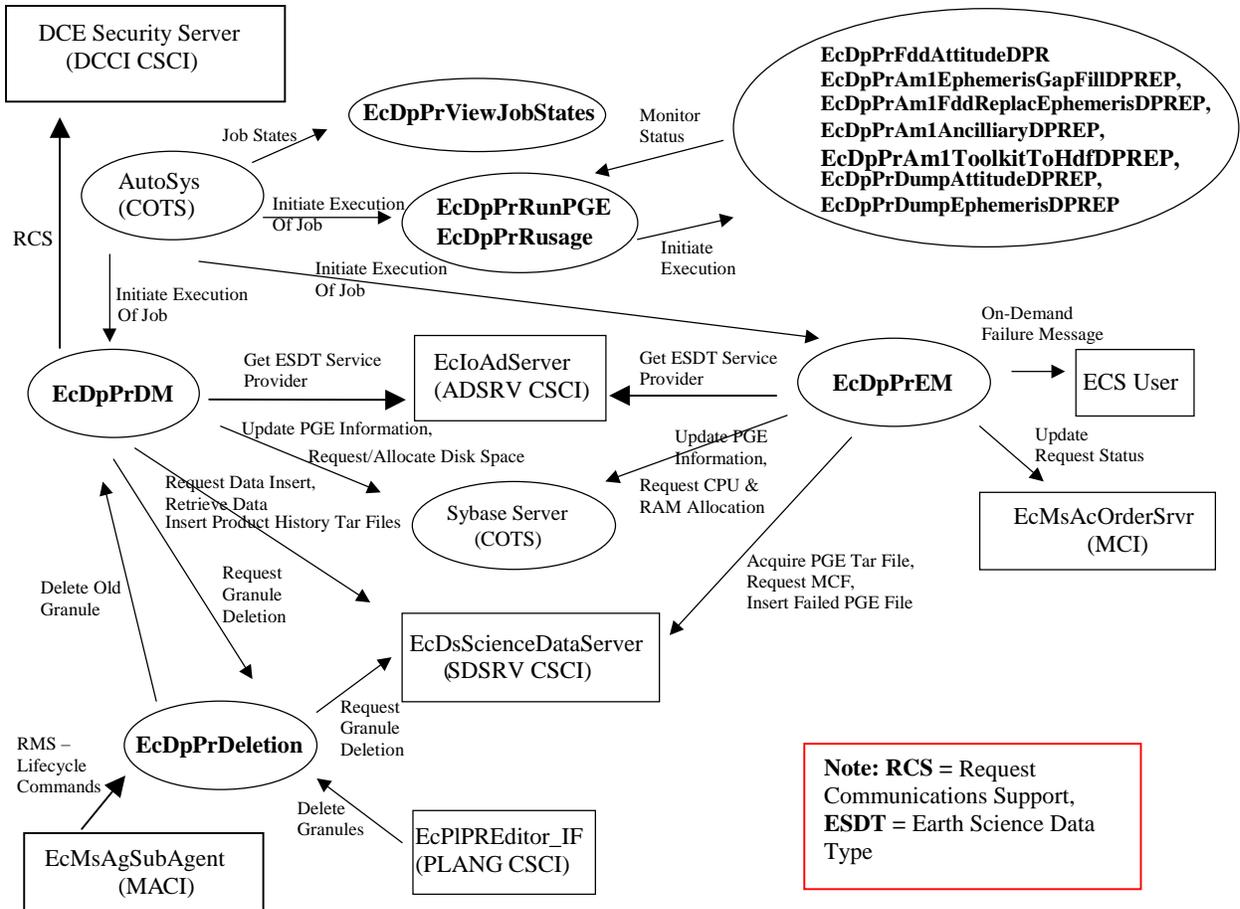


Figure 4.7.1.3-2. PRONG CSCI Architecture Diagram (cont.)

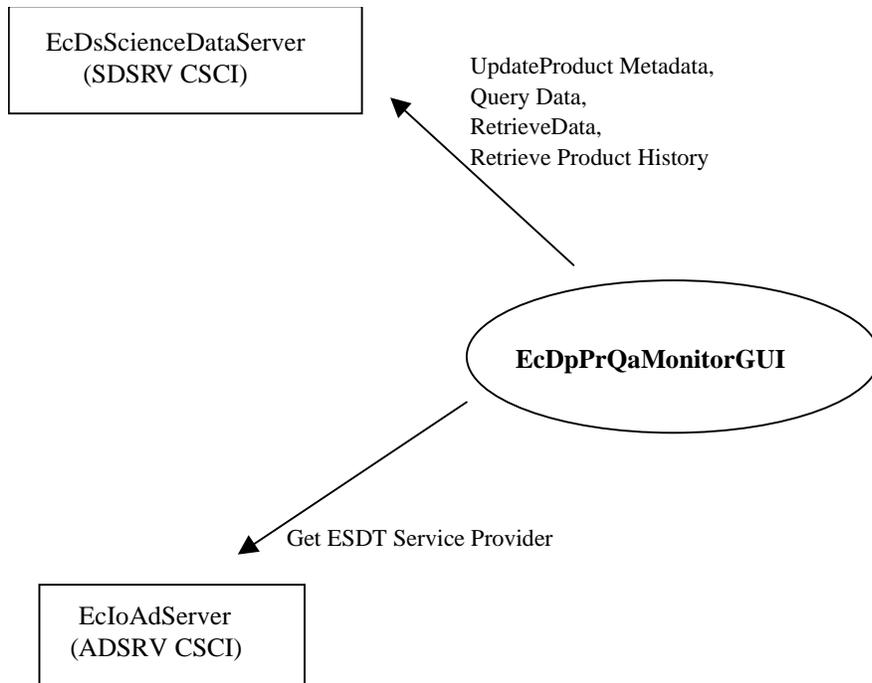


Figure 4.7.1.3-3. PRONG CSCI Architecture Diagram (cont.)

4.7.1.4 Processing Process Descriptions

Table 4.7.1.4-1 provides descriptions of the processes shown in the PRONG CSCI architecture diagrams. **Note:** Resource Management is an integral part of PRONG. Unlike the Data Manager or Execution Manager, it is a library of routines used by them instead of being executed like them. The Resource Manager checks the resource requirements (i.e., disk space, memory, and Central Processing Unit (CPU)) of a PGE during processing, determines the availability of those resources, and dedicates their usage until PGE execution finishes.

Table 4.7.1.4-1. PRONG CSCI Processes (1 of 4)

Process	Type	COTS/ Developed	Functionality
EcDpPrDM	Other	Developed	The Data Management process manages the flow of science data to and from science processing resources including communication mechanisms to interface with the EcDsScienceDataServer. Data Management manages data retention on science processing resources to support PGE executions.
EcDpPrEM	Other	Developed	The Execution Management process initiates the execution of PGEs (via the COTS product AutoSys). EcDpPrEM supports the preparation activities prior to the execution of PGEs and subsequent activities to the execution of PGEs. EcDpPrEM also provides status on On-Demand Processing Requests and send out e-mail to the originator in the event of a failure.
EcDpPrRunPGE EcDpPrRusage	Other	Developed	The PGE Execution Manager process controls and monitors PGE executions including Process Control File creation and output product storage growth. EcDpPrRusage measures the actual resources used by the PGE and reports to AutoSys unexpected resource usage.
EcDpPrDeletion	Server	Developed	This DCE Server notifies the EcDsScienceDataServer to remove interim granules via the data management process (EcDpPrDM) when they are no longer needed. The interim products are removed after the last PGE in the chain has used them or a pre set time has expired after the last use of the interim product. It also is used by the PLS to delete granules associated with a cancelled DPR.

Table 4.7.1.4-1. PRONG CSCI Processes (2 of 4)

Process	Type	COTS/ Developed	Functionality
AutoSys	GUI	COTS	<p>AutoSys is a job scheduling software application used to automate operations in a distributed UNIX environment. AutoSys executes jobs to automate support for PGE execution. AutoSys creates job boxes consisting of a series of related jobs, and manages job dependencies. AutoSys provides graphical depictions of completed jobs and jobs being processed. It includes the Operator Console GUI to allow human intervention into monitoring and altering the AutoSys job stream. The daily job schedule is submitted to the Job Management server at the start of the processing day. Jobs, which have data available, are released into AutoSys. To support job executions, AutoSys requires additional help for:</p> <ul style="list-style-type: none"> • Allocation of sufficient resources (e.g., disk space) to support executions. The EcDpPrDM provides the capabilities to manage disk space and monitor resources • Managing remote host data acquisition, data retention on the DPS processing host, and data distribution from the DPS processing host • Initialization and PGE executions.

Table 4.7.1.4-1. PRONG CSCI Processes (3 of 4)

Process	Type	COTS/ Developed	Functionality
<p>EcDpPrJobMgmt EcDpPrJobMgmtClient EcDpPrViewJobStates</p>	<p>Server</p>	<p>Developed</p>	<p>The Job Management process uses the AutoSys COTS product to create and initiate execution of PRONG administrative jobs for managing SPRHW assets and for PGE execution. Seven Unix processes bundled together into an AutoSys job box perform this work. Job Management is responsible for efficient AutoSys management so the maximum number of jobs possible can be continuously run using the product. This involves controlling the flow of jobs through AutoSys by only allowing jobs ready to run into the product and by removing jobs as they complete. Job Management also creates and starts execution of Ground Event jobs in AutoSys.</p> <p>The Job Management Client process is used by programs that need access to the Job Management Server services to modify jobs in AutoSys to change the priority of the jobs.</p> <p>The various events this process provides are: <u>CreateDPR</u>: A data processing request identified is then translated into seven standard process steps (one, the PGE execution, the remaining performing support activities). If the science data is available, the job box containing the seven individual jobs is released into AutoSys. <u>ReleaseDPR</u>: A previously created data processing request waiting for the availability of science data is released into AutoSys to begin execution. <u>CancelDPR</u>: This provides the capability to cancel/terminate a data processing request. <u>CreateGEvntJob</u>: Create a Ground Event Job in AutoSys. <u>CancelGEvntJob</u>: Cancel a Ground Event Job.</p> <p>The View Job States process allows the Operations Staff to view jobs in the queue to determine the completed jobs, the jobs executing, and the jobs awaiting execution.</p>

Table 4.7.1.4-1. PRONG CSCI Processes (4 of 4)

Process	Type	COTS/ Developed	Functionality
EcDpPrQaMonitorGUI	GUI	Developed	This process provides the capability to transfer science data from the archives, browse data images, and examine and update science metadata. It is an automated tool for performing data analysis in support of DAAC Quality Assurance activities.
EcDpPrAm1AncillaryDPREP, EcDpPrAm1EphemerisGapFillDPREP, EcDpPrFddAttitudeDPREP, EcDpPrAm1FddReplaceEphemerisDPREP, EcDpPrAm1ToolkitToHdfDPREP, EcDpPrDumpAttitudeDPREP, EcDpPrDumpEphemerisDPREP	Other	Developed	Data Preprocessing manages L0 attitude and ephemeris ancillary data preprocessing for inputs to PGEs. Data preprocessing is the preliminary processing or application of an operation on a data set that does not alter or modify scientific content of the data set. Preprocessing includes data set format changes by reordering the lower level byte structure, data set reorganization (ordering data items within and between physical files), and preparing additional metadata based on lower level metadata. PM1 Orbit Processing includes reformatting the FDD definitive and predictive ephemeris data sets into the toolkit native format and the HDF format, respectively. Also, generates the orbit metadata records for the data sets.
Sybase Server	Server	COTS	The Sybase Server acts as a SQL server for the PDPS database.
EcDpPrGE	Other	Developed	The EcDpPrJobMgmt server initiates the EcDpPrGE when the server gets a ground event request. The ground event process starts at a specified time and runs a specified duration. During the time the ground event process runs, it sets a computer resource (cpu, ram, etc.) off-line and the computer resource is not available for PGEs.

4.7.1.5 Processing Process Interface Descriptions

Tables 4.7.1.5-1, 4.7.1.5-2, and 4.7.1.5-3 provide descriptions of the interface events shown in the three respective PRONG CSCI architecture diagrams.

Table 4.7.1.5-1. PRONG CSCI Process Interface Events (1 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
DPR Control, Status & QA	Per data processing request	<i>Process:</i> EcDpPrJobMgmt <i>Library:</i> DpPrJM <i>Class:</i> DpPrScheduler	Operations Staff/ Operations Staff terminal	The Operations staff controls Data Processing Request (DPR) activity with the capability to cancel, suspend, resume, and modify a DPR. The M&O staff supports status collecting, PRONG hardware resource monitoring and Quality Assurance validating processes.
View Job States	Per Operations Staff request.	<i>Process:</i> EcDpPrViewJobStates <i>Class:</i> DpPrListJobs	Operations Staff/ Operations Staff terminal	The Operations staff can view the job state via the EcDpPrViewJobStates process, as an aid in scheduling jobs.
Create Ground Event Job	One per defined ground event	<i>Process:</i> AutoSys (COTS)	<i>Process:</i> EcDpPrJobMgmt <i>Library:</i> DpPrJM <i>Class:</i> DpPrScheduler	The EcDpPrJobMgmt process sends a request to AutoSys to create a ground event job to perform maintenance activities on data processing resources.
Cancel Ground Event Job	One per defined ground event	<i>Process:</i> AutoSys (COTS)	<i>Process:</i> EcDpPrJobMgmt <i>Library:</i> DpPrJM <i>Class:</i> DpPrScheduler	The EcDpPrJobMgmt process sends a request to AutoSys to cancel a ground event job for the deletion of a ground event.
Create Job Box	One per DPR	<i>Process:</i> AutoSys (COTS)	<i>Process:</i> EcDpPrJobMgmt <i>Library:</i> DpPrJM <i>Class:</i> DpPrScheduler	The EcDpPrJobMgmt sends a request to AutoSys to create a job box when all DPR input data is available using Job Interface Language (JIL). If all the input data is not available, the EcDpPrJobMgmt stores the DPR in a priority-based queue.
Release Job Box	One per execution of a DPR in the jobmgmt queue	<i>Process:</i> AutoSys (COTS)	<i>Process:</i> EcDpPrJobMgmt <i>Library:</i> DpPrJM <i>Class:</i> DpPrScheduler	The EcDpPrJobMgmt sends a request to release (or delete) a job box in the AutoSys queue for a DPR awaiting execution. <i>Process:</i> EcPISubMgr <i>Class:</i> PISubscriptionManagerProcess

Table 4.7.1.5-1. PRONG CSCI Process Interface Events (2 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
Cancel Job Box	One per job box deletion	<i>Process:</i> AutoSys (COTS)	<i>Process:</i> EcDpPrJobMgmt <i>Library:</i> DpPrJM <i>Class:</i> DpPrScheduler	The EcDpPrJobMgmt deletes a job box in AutoSys and performs any cleanup when an operator requests a DPR to be canceled.
Update PGE Information	One per update/retrieve data request	Sybase Database (COTS)	<i>Processes:</i> EcDpPrJobMgmt, EcDpPrJobMgmtClient <i>Library:</i> DpPrRM	The EcDpPrJobMgmt and EcDpPrJobMgmtClient processes, using the DpPrRM library, request update and retrieval of data in the database that defines a PGE and allows the PGE to be scheduled and executed by AutoSys. These processes request updates for granule information (location, size, etc.), processing status, and check-pointing information stored in the database.
Manage DPRs (create, release, cancel)	One per control request	<i>Process:</i> EcDpPrJobMgmt <i>Library:</i> DpPrJM <i>Class:</i> DpPrScheduler	<i>Processes:</i> EcPIWb, EcPISubMgr, EcPIRpSi <i>Library:</i> PICore1 <i>Classes:</i> PIWbScheduler, PIDpr	The EcPIWb process sends requests to the EcDpPrJobMgmt to create and cancel (delete) DPR jobs in AutoSys. The EcPISubMgr process sends requests to the EcDpPrJobMgmt process to release DPR jobs in AutoSys.
Create Ground Event	One per ground resource	<i>Process:</i> EcDpPrJobMgmt <i>Library:</i> DpPrJM <i>Class:</i> DpPrScheduler	<i>Process:</i> EcPIWb <i>Class:</i> PIWbScheduler	The EcPIWb sends requests to the EcDpPrJobMgmt process to create ground events for maintenance activities on data processing resources
Cancel Ground Event	One per ground resource	<i>Process:</i> EcDpPrJobMgmt <i>Library:</i> DpPrJM <i>Class:</i> DpPrScheduler	<i>Process:</i> EcPIRpSi <i>Class:</i> PISiMain	The EcPIRpSi process sends a request to the EcDpPrJobMgmt process to cancel ground events in AutoSys.

Table 4.7.1.5-1. PRONG CSCI Process Interface Events (3 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
Request Management Services	One per process or server	<i>Processes:</i> EcDpPrJobMgmt, EcDpPrJobMgmtClient	<i>Process:</i> EcMsAgSubAgent <i>Library:</i> EcAgInstrm <i>Class:</i> EcAgManager	The MACI provides a basic management library of services to the CSCIs, implemented as client or server applications, using the CSS Process Framework. The basic management library of services include: <ul style="list-style-type: none"> • Lifecycle Commands –The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training).
Initiate execution (control)	One per PGE job execution	<i>Process:</i> EcDpPrGE <i>Class:</i> DpPrGE	<i>Process:</i> EcDpPrJobMgmt <i>Library:</i> DpPrJM <i>Class:</i> DpPrScheduler	The EcDpPrJobMgmt process initiates the EcDpPrGE process when ground events occur.

Table 4.7.1.5-2. PRONG CSCI Process Interface Events (1 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
Job States	Per AutoSys Status update.	<i>Process:</i> EcDpPrViewJobStates <i>Class:</i> DpPrListJobs	<i>Process:</i> AutoSys (COTS)	The AutoSys provides the job state (completed, executing, or in a queue to be executed) to the EcDpPrViewJobStates process.
Initiate execution of job (control)	One per PGE job execution	<i>Process:</i> EcDpPrDM <i>Library:</i> EcDpPrDM <i>Class:</i> DpPrDm <i>Process:</i> EcDpPrEM <i>Library:</i> DpPrEM <i>Class:</i> DpPrExecutionManagement <i>Process:</i> EcDpPrRunPGE <i>Class:</i> DpPrRunPGE	<i>Process:</i> AutoSys (COTS)	AutoSys initiates the execution of the EcDpPrDM, the EcDpPrEM and the EcDpPrRunPGE processes to control the preparation (data staging), execution, and archiving of higher level products, which are produced, and cleanup of each PGE run.
Monitor status (status)	One per PGE job execution	<i>Processes:</i> EcDpPrAm1AncillaryDPREP, EcDpPrAm1EphemerisGapFillDPREP, EcDpPrFddAttitudeDPREP, EcDpPrAm1FddReplaceEphemerisDPREP, EcDpPrAm1ToolkitToHdfDPREP, EcDpPrDumpAttitudeDPREP, EcDpPrDumpEphemerisDPREP	<i>Process:</i> EcDpPrRunPGE <i>Class:</i> DpPrRunPGE	The EcDpPrRunPGE process, apart from initiating the PGE process, also monitors the PGE's computer resources. If the PGE's computer resources exceed its expected usage an alarm is sent to the AutoSys. This wrapper also captures the PGE's resource its exit code usage and.

Table 4.7.1.5-2. PRONG CSCI Process Interface Events (2 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
Initiate execution (control)	One per PGE job execution	<i>Processes:</i> EcDpPrAm1AncillaryDPREP, EcDpPrAm1EphemerisGapFillDPREP, EcDpPrFddAttitudeDPREP, EcDpPrAm1FddReplaceEphemerisDPREP, EcDpPrAm1ToolkitToHdfDPREP, EcDpPrDumpAttitudeDPREP, EcDpPrDumpEphemerisDPREP	<i>Process:</i> EcDpPrRunPGE <i>Class:</i> DpPrRunPGE	The EcDpPrRunPGE provides a buffer between AutoSys and the PGE. This serves as a wrapper to the PGE process, initiates the PGE execution and captures the PGE's exit status.
On-Demand Failure Message	Only in the event of On-Demand Processing Request failure	<i>User</i> <i>Library:</i> CsEmMailRelA <i>Class:</i> CsEmMailRelA	<i>Process:</i> EcDpPrEM <i>Library:</i> DpPrEM <i>Class:</i> DpPrExecutionManagement	The EcDpPrEM process sends an e-mail message to the originator of an On-Demand Processing Request if any DPR generated by that request fails.
Update Request Status	One for each PGE run to satisfy the requested On-Demand Product	<i>Process:</i> EcAcOrderSvr <i>Libraries:</i> EcAcSrv, EcAcComm <i>Class:</i> EcAcOrderManager	<i>Process:</i> EcDpPrEM <i>Library:</i> DpPrEM <i>Class:</i> DpPrExecutionManagement	The EcDpPrEM process updates the status of an On-Demand Processing Request when a DPR for the order completes or fails processing.

Table 4.7.1.5-2. PRONG CSCI Process Interface Events (3 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
Get ESDT Service provider	One per Advertising search	<i>Process:</i> EcIoAdServer <i>Libraries:</i> IoAdcore, IoAdSubs <i>Classes:</i> IoAdSignatureServiceAdv, IoAdApprovedAdv, IoAdGroup, IoAdProvider	<i>Processes:</i> EcDpPrEM, EcDpPrDM <i>Library:</i> PICore1 <i>Class:</i> PIDataType	The EcDpPrDM and EcDpPrEM send requests to the EcIoAdServer, using the Universal Reference obtained from the EcDsScienceDataServer, for a particular ESDT.
Update PGE Information	One per update/retrieve data request	Sybase Database (COTS)	<i>Processes:</i> EcDpPrDM, EcDpPrEM <i>Library:</i> DpPrRM	The EcDpPrDM, and EcDpPrEM processes, using the DpPrRM library, request update and retrieval of data in the database that defines a PGE and allows the PGE to be scheduled and executed by AutoSys. These processes request updates for granule information (location, size, etc.), processing status, and check-pointing information stored in the database.
Request CPU and RAM Allocation	One per PGE job execution	Sybase Server (COTS)	<i>Process:</i> EcDpPrEM <i>Library:</i> DpPrRM <i>Class:</i> DpPrResourceManager	The EcDpPrEM process requests CPU and RAM allocations via the Sybase Server by using the DpPrRM library software for each PGE based on values entered at SSIT.
Acquire PGE Tar file	One per tar file acquire	<i>Process:</i> EcDsScienceData Server <i>Library:</i> DsCI <i>Classes:</i> DsCIRequest, DsCICommand	<i>Process:</i> EcDpPrEM <i>Library:</i> DpPrDssIF <i>Class:</i> DpPrDSSInterface	The EcDpPrEM acquires a tar file for any PGE not currently local to the science processor from the EcDsScienceDataServer. The tar file is removed from the tape archive and used during PGE execution.
Request MCF	One per MCF request	<i>Process:</i> EcDsScienceData Server <i>Library:</i> DsCI <i>Classes:</i> DsCIDescriptor	<i>Process:</i> EcDpPrEM <i>Library:</i> DpPrDssIF <i>Class:</i> DpPrDSSInterface	The EcDpPrEM requests a MCF template for each output data type for specific PGEs from the EcDsScienceDataServer and the MCF template is populated with metadata from the output granule.

Table 4.7.1.5-2. PRONG CSCI Process Interface Events (4 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
Insert Failed PGE Tar file	One per unsuccessful PGE execution	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsCI <i>Classes:</i> DsCIRequest, DsCICommand	<i>Process:</i> EcDpPrEM <i>Library:</i> DpPrDssIF <i>Class:</i> DpPrDSSInterface	After an unsuccessful execution of a PGE, the EcDpPrEM obtains the Tar file containing the PGE log files, core dump (if any), PCF and other files, and requests the files be inserted into the EcDsScienceDataServer for permanent archive.
Request Granule Deletion	Per granule delete request	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsCI <i>Classes:</i> DsCIRequest, DsCICommand, DsCIESDTReferenceCollector	<i>Process:</i> EcDpPrDeletion <i>Class:</i> DpDeletion	The EcDpPrDeletion process submits the request to delete interim granules a PGE had used in processing or after a defined storage period has elapsed to the EcDsScienceDataServer.
Delete Granules	Per granule delete request	<i>Process:</i> EcDpPrDeletion <i>Class:</i> DpDeletion	<i>Process:</i> EcPIPReDitor_IF <i>Library:</i> PICore1 <i>Classes:</i> PIWbScheduler, PIDpr	The EcPIPReDitor_IF sends requests to the EcDpPrDeletion process to delete granules associated with a DPR, which has been cancelled.

Table 4.7.1.5-2. PRONG CSCI Process Interface Events (5 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
Request Management Services	Per applications need.	<i>Process:</i> EcDpPrDeletion	<i>Process:</i> EcMsAgSubAgent <i>Library:</i> EcAgInstrm <i>Class:</i> EcAgManager	The EcMsAgSubAgent process provides a basic management library of services to the processes, implemented as client or server applications, using the DCCI CSCI Process Framework. The basic management library of services include: <ul style="list-style-type: none"> • Lifecycle Commands – The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training).
Request Granule Deletes	Per granule delete request	<i>Process:</i> EcDpPrDeletion <i>Class:</i> DpDeletion	<i>Process:</i> EcDpPrDM <i>Library:</i> DpPrDssIF <i>Class:</i> DpPrDSSInterface	The EcDpPrDM sends requests to the EcDpPrDeletion process to delete interim granules a PGE had used in processing or after a defined storage period has elapsed. The EcDpPrDeletion process submits the request to the EcDsScienceDataServer.
Delete Old Granule	One per granule	<i>Process:</i> EcDpPrDM <i>Library:</i> DpPrDM <i>Class:</i> DpPrGranuleLocator	<i>Process:</i> EcDpPrDeletion <i>Class:</i> DpDeletion	The EcDpPrDeletion process gets a request from the EcPIWb, via the EcDpPrDM, to delete a granule from one of the local science processing disks. The EcDpPrDeletion process uses the DpPrDM library software to delete the files from the disk and update the PDPS database.

Table 4.7.1.5-2. PRONG CSCI Process Interface Events (6 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
Request/Allocate Disk Space	One per disk request	Sybase Server (COTS)	<i>Process:</i> EcDpPrDM <i>Library:</i> DpPrRM <i>Class:</i> DpPrResourceManager	The EcDpPrDM requests disk space via the Sybase Server by using the DpPrRM library software for each input granule that needs to be staged to the local processing disk and output granule needed by a PGE.
Request Data Insert	One granule per request	<i>Process:</i> EcDsScienceData Server <i>Library:</i> DsCI <i>Classes:</i> DsCIRequest, DsCICommand, DsCIESDTReferenceCollector	<i>Process:</i> EcDpPrDM <i>Library:</i> DpPrDssIF <i>Class:</i> DpPrDSSInterface	After the PGE has successfully completed executing, the EcDpPrDM sends insert requests for the EcDsScienceDataServer to store the output granules into the SDSRV inventory/archives.
Retrieve Data	One granule per request	<i>Process:</i> EcDsScienceData Server <i>Library:</i> DsCI <i>Classes:</i> DsCIRequest, DsCICommand, DsCIESDTReferenceCollector	<i>Process:</i> EcDpPrDM <i>Library:</i> DpPrDssIF <i>Class:</i> DpPrDSSInterface	The EcDpPrDM acquires the input granules needed by a PGE not currently local on a science processor from the EcDsScienceDataServer.
Insert Product History Tar files	One per successful PGE execution	<i>Process:</i> EcDsScienceData Server <i>Library:</i> DsCI <i>Classes:</i> DsCIRequest, DsCICommand	<i>Process:</i> EcDpPrDM <i>Library:</i> DpPrDssIF <i>Class:</i> DpPrDSSInterface	After the PGE has successfully completed executing and archiving the resulting outputs, the EcDpPrDM requests the PGE Production History Tar file be inserted into the EcDsScienceDataServer for permanent archive.

Table 4.7.1.5-2. PRONG CSCI Process Interface Events (7 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
Request Communications Support	One service per request.	<p>Process: DCE Security Server</p> <p>Libraries: EcSeLogin, EcSeLogincontext</p> <p>Classes: EcSeLogin, EcSeLogincontext</p> <p>Library: EcPf</p> <p>Classes: EcPfManagedServer, EcPfclient</p> <p>Library(Common): EcUr</p> <p>Class: EcUrServerUR</p> <p>Process:</p> <p>Library: Event</p> <p>Class: EcLgErrormsg</p>	<p>Process: EcDpPrDM</p> <p>Library: DpPrDM</p> <p>Class: DpPrDataManager</p>	<p>The DCCI CSCI Process Framework provides a library of services available to each SDPS and CSMS process. The services required to perform the specific process assignments are requested by the process from the Process Framework. These services include: DCE support, file transfer services, Network & Distributed File Services, Bulk Data transfer services, file copying services, name/address services, password services, Server Request Framework (SRF), UR, Error/Event logging, message passing, Fault Handling services, User Authentication services, and Mode information.</p>

Table 4.7.1.5-3. PRONG CSCI Process Interface Events (1 of 2)

Event	Event Frequency	Interface	Initiated By	Event Description
Update Product Metadata	One per metadata product update	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsCI <i>Classes:</i> DsCICommand, DsCIRequest, DsCIESDTReferenceCollector	<i>Process:</i> EcDpPrQaMonitorGUI <i>Library:</i> DpPrQaMonitor <i>Class:</i> DpPrQAGranuleQaFlags	The EcDpPrQaMonitorGUI provides the operator with capabilities to update product metadata in the EcDsScienceDataServer.
Query Data	One per query	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsCI <i>Class:</i> DsCIESDTReferenceCollector	<i>Process:</i> EcDpPrQaMonitorGUI <i>Library:</i> DpPrQaMonitor <i>Class:</i> DpPrQaDataGranule	The EcDpPrQaMonitorGUI submits requests of this type to the EcDsScienceDataServer. It searches the archive for granules that match the user-supplied selection criteria: data type and begin/end date. Results are displayed to the user.
Retrieve Data	One per request	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsCI <i>Class:</i> DsCIAcquireCommand	<i>Process:</i> EcDpPrQaMonitorGUI <i>Library:</i> DpPrQaMonitor <i>Class:</i> DpPrQaMonitor	The EcDpPrQaMonitorGUI submits requests of this type to the EcDsScienceDataServer. It transfers a granule from the Science Data archive to the user's host machine.
Retrieve Product History	One per request	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsCI <i>Class:</i> DsCIAcquireCommand	<i>Process:</i> EcDpPrQaMonitorGUI <i>Library:</i> DpPrQaMonitor <i>Class:</i> DpPrQaMonitor	The EcDpPrQaMonitorGUI submits requests of this type to the EcDsScienceDataServer. It transfers the Production History tar file from the Science Data archive to the user's host machine.

Table 4.7.1.5-1. PRONG CSCI Process Interface Events (2 of 2)

Event	Event Frequency	Interface	Initiated By	Event Description
Get ESDT Service provider	One per Advertising search	<i>Process:</i> EcloAdServer <i>Libraries:</i> IoAdcore, IoAdSubs <i>Classes:</i> IoAdSignatureServiceAdv, IoAdApprovedAdv, IoAdGroup, IoAdProvider	<i>Process:</i> EcDpPrQaMonitorGUI <i>Library:</i> PICore1 <i>Class:</i> PIDataType	The EcDpPrQaMonitorGUI sends requests to the EcloAdServer, using the Universal Reference obtained from the EcDsScienceDataServer, for a particular ESDT.

4.7.1.6 Processing Data Stores

Table 4.7.1.6-1 provides descriptions of the data stores shown in the PRONG CSCI architecture diagram.

Table 4.7.1.6-1. PRONG CSCI Data Stores

Data Store	Type	Functionality
PDPS database	Database	The PDPS database is replicated within the same site and holds the persistent data for PDPS. The persistent data includes, but is not limited to, resource information entered with the Resource Planning utilities, PGE and data type information entered at SSIT, Production Requests, Data Processing Requests and Data Granule information entered using the Production Request Editor and plan information entered using the Production Planning Workbench.

4.7.2 Algorithm Integration and Test Tools Software Description

4.7.2.1 Functional Overview

The Algorithm Integration and Test Tools (AITTL) are used by the DAAC Integration and Test (I&T) team to:

- Retrieve science software and submit it for configuration control
- Compile and link the delivered source files
- Execute test cases
- Provide error diagnosis using interactive debuggers, and data viewers
- Collect resource metrics of CPU time, memory, and disk space to build the PGE Profile and thus enable the PLANG and PRONG CSCIs to execute the science software
- Update the system databases after the science software completes acceptance testing

The AITTL tools are in the following categories:

- Compilers, linkers, debuggers, and other development and operating system tools
- Tools for viewing science software documentation
- Tools for checking compliance of science software to Earth Science Data and Information System (ESDIS)-specified coding standards
- Code analysis tools (e.g., Sparc Works, CASEVision)
- Data viewing tools (e.g., EOSView).
- Tools for comparing HDF files
- Tools for comparing Binary files
- Tools for providing executable profiles (to get a PGE performance profile)
- Tools to register the science software with the Planning and Data Processing Subsystems
- Tools to add and update Science Software Archive Packages (SSAPs) in the Data Server
- Tools for writing reports and maintaining the I&T logs
- Tools for checking Process Control Files and for prohibited functions
- Tools to display product metadata

For information on science software integration and test procedures, see Science User's Guide and Operations Procedures Handbook for the ECS Project (205-CD-002-001) Part 4, and Science Software Integration and Test (JU9403V1). For information on the ESDIS science software coding standards and guidelines, see Data Production Software and Science Computing Facility (SCF) Standards and Guidelines (423-16-01).

Note: The directory structure for the AITTL software has the name SSIT and not AITTL. The use of the SSIT directory structure name is to denote the main purpose of the Algorithm Test Tools as tools to support the Science Software Integration and Test activities as part of the SDPS data processing.

4.7.2.2 Algorithm Integration and Test Tools Context

Figure 4.7.2.2-1 is the AITTL CSCI context diagram. The diagram shows the events sent to the AITTL CSCI and the events the AITTL CSCI sends to other ECS subsystems.

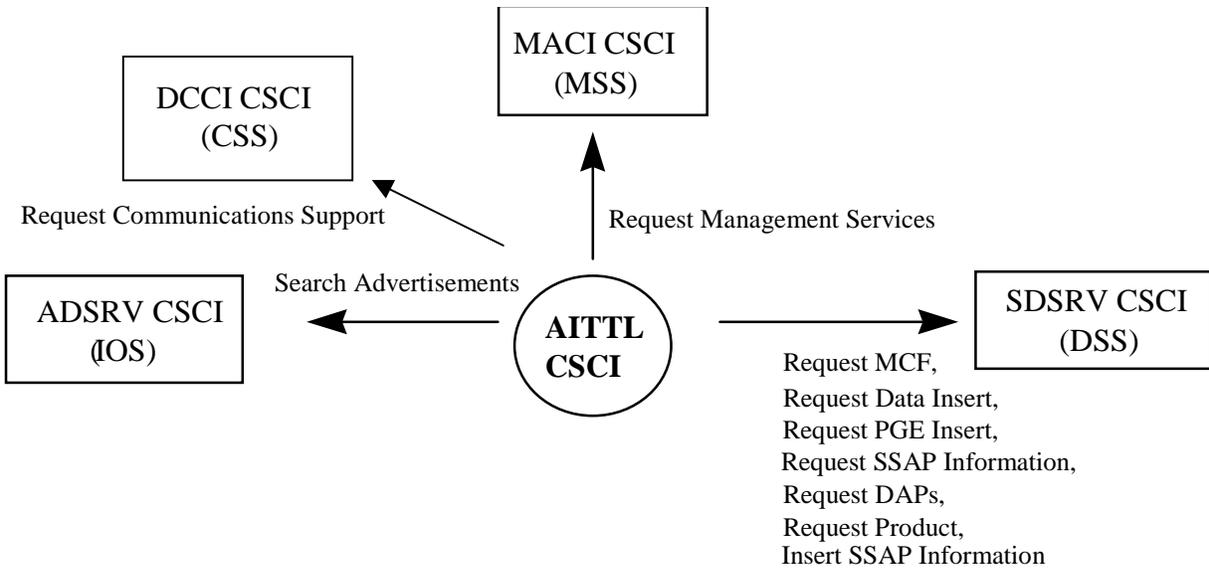


Figure 4.7.2.2-1. AITTL Context Diagram

Table 4.7.2.2-1 provides descriptions of the interface events shown in the AITTL Context Diagram.

Table 4.7.2.2-1. AITTL Interface Events (1 of 2)

Event	Interface Event Description
Request Management Services	<p>The MACI provides a basic management library of services to the CSCIs, implemented as client or server applications, using the CSS Process Framework. The basic management library of services include:</p> <ul style="list-style-type: none"> Lifecycle commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training).
Request MCF	<p>The AITTL CSCI sends requests to the SDSRV CSCI for the MCF template for use during SSIT. The PRONG CSCI also requests the MCF template from the SDSRV CSCI prior to a data insert request.</p>
Request Data Insert	<p>The AITTL CSCI puts various types of data into the SDSRV inventory, from SSAP information to Static files and PGE executables. In response, the AITTL CSCI gets the results of the insert and the UR to perform an acquire request.</p>
Request PGE Insert	<p>The AITTL CSCI sends requests to insert data that defines a PGE and allows it to be scheduled and executed.</p>

Table 4.7.2.2-1. AITTL Interface Events (2 of 2)

Event	Interface Event Description
Request SSAP Information	The AITTL CSCI sends requests to the SDSRV CSCI for SSAP information, including names of existing SSAPs and the information associated with a specific SSAP. In response, the SDSRV CSCI sends lists of SSAPs and related information.
Request DAPs	The AITTL CSCI requests DAPs based on URs from the SDSRV CSCI. The requested DAPs are placed on a local AITTL disk.
Request Product	The AITTL CSCI sends requests, to the SDSRV CSCI, for particular data granules to be pushed, via the FTP service, onto the DPS science processor as input for data processing or for SSIT work.
Insert SSAP Information	The M&O Staff sends requests to the SDSRV CSCI to insert SSAP information, via the SSAP GUI, including SSAP name, SSAP version number, PGE name, PGE version number, and SSAP Acceptance Date.
Search Advertisements	The ADSRV CSCI receives search requests for subscription event and signature service advertisements from the AITTL CSCI. The AITTL CSCI obtains the proper signatures for acquiring and inserting data granules from/to the SDSRV CSCI.
Request Communications Support	The DCCI CSCI provides a library of services available to each SDPS and CSMS CSCI. The services required to perform the specific CSCI assignments are requested by the CSCI from the DCCI CSCI. These services include: DCE support, file transfer services, Network & Distributed File Services, Bulk Data transfer services, file copying services, name/address services, password services, Server Request Framework (SRF), UR, Error/Event logging, message passing, Fault Handling services, User Authentication services, Mode information, and retrieving the requested configuration attribute-value pairs from the Configuration Registry for ECS applications that request them.

4.7.2.3 Algorithm Integration and Test Tools Architecture

Figures 4.7.2.3-1 and 4.7.2.3-2 are the AITTL CSCI architecture diagrams. The diagrams show the events that launch the AITTL CSCI processes and the events the AITTL CSCI processes send to processes in other CSCIs.

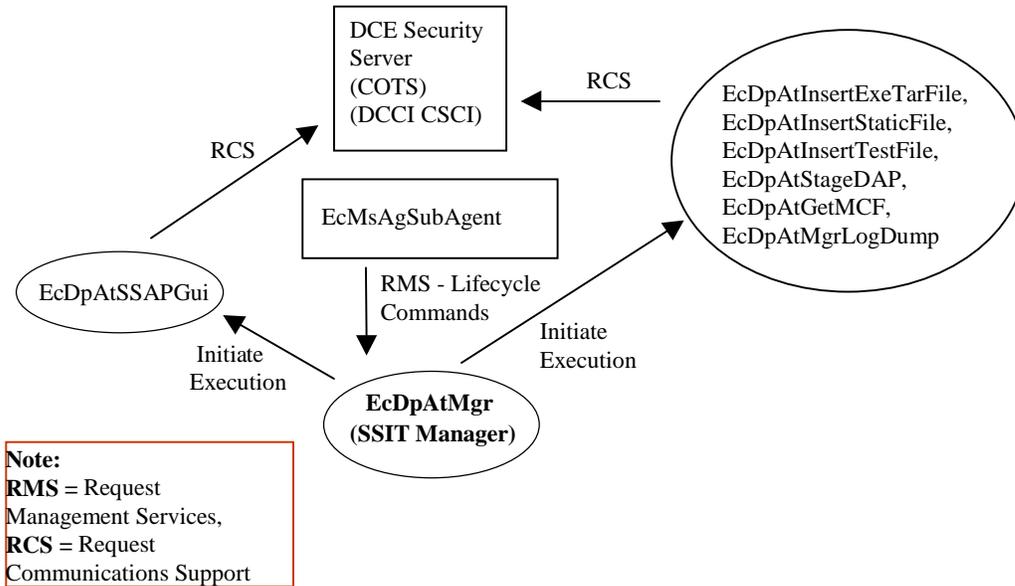


Figure 4.7.2.3-1. AITTL CSCI Architecture Diagram

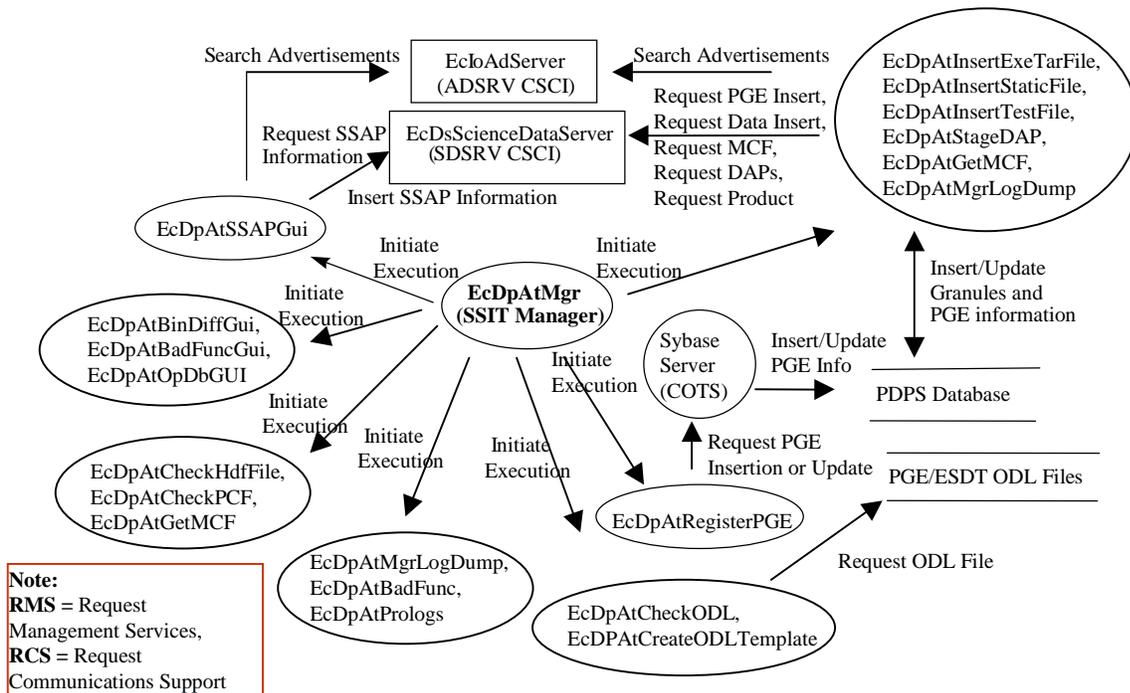


Figure 4.7.2.3-2. AITTL CSCI Architecture Diagram (cont.)

4.7.2.4 Algorithm and Test Tools Process Description

Table 4.7.2.4-1 provides descriptions of the processes shown in the AITTL CSCI architecture diagram.

Table 4.7.2.4-1. AITTL Processes (1 of 2)

Process	Type	COTS / Developed	Functionality
EcDpAtSSAPGui	GUI	Developed	This GUI allows the M&O staff to create, update and delete SSAPs and to acquire information about an SSAP for modification or testing (such as test plans).
EcDpAtMgr	GUI	Developed	This application provides menus to launch other SSIT applications and provides a checklist to users for marking each SSIT function as completed.
EcDpAtRegisterPGE, EcDpAtCheckODL, EcDpAtCreateODLTemplate, EcDpAtOpDbGui	GUI and cmd line interface (I/F)	Developed	This application group allows a PGE to be defined in the PDPS database. ODL is read and checked by the tools and translated into the fields defining a PGE in the PDPS database. If the ODL files are valid, each row already existing in the PDPS database is updated and non-existent rows are inserted. The SSIT personnel input performance information via a GUI.
EcDpAtBinDiffGui, hdiff	GUI and cmd line I/F	Developed and COTS	This application group supports data file viewing and comparisons. The group includes EOSView, the COTS language IDL, and tools to compare binary and HDF files. The shell programs EcDpAtCheckhdfFile and EcDpAtMgrXdiff are used to assist with the viewing and comparisons.
EcDpAtBadFunc, EcDpAtBadFuncGui, EcDpAtPrologs, EcDpAtCheckPCF	GUI	Developed and COTS	This applications group checks the source code for PGEs and PGE PCFs for errors or prohibited functions. The Sparc Works COTS product is included for editing and debugging functions and a checker is provided for use during testing. Also provided is a checker to monitor the software for any prohibited calls
EcDpAtInsertExeTarFile, EcDpAtInsertStaticFile, EcDpAtInsertTestFile, EcDpAtStageDAP, EcDpAtGetMCF	cmd line I/F	Developed	This applications group provides mechanisms to insert and acquire data items from the EcDsScienceDataServer in the SDSRV CSCI. Static Files and PGE executables are inserted into the SDSRV archives by these tools, and the respective PDPS database tables are updated with the results. The Delivery Archive Package (DAP) and MCFs are acquired via these tools for command line testing the PGE.

Table 4.7.2.4-1. AITTL Processes (2 of 2)

Process	Type	COTS / Developed	Functionality
EcDpAtMgrLogDump	Cmd line I/F	Developed	Command line interface to dump the SSIT checks list database to a file that can be sent to the printer.
Sybase Server	Server	COTS	The Sybase Server is the interface between AITTL processes and the PDPS database for PGE insertion and update of PGE information in the PDPS database to support Data Processing activities.

4.7.2.5 Algorithm and Test Tools Process Interface Descriptions

Tables 4.7.2.5-1 and 4.7.2.5-2 provide descriptions of the interface events shown in the AITTL CSCI architecture diagrams.

Table 4.7.2.5-1. AITTL Process Interface Events (1 of 2)

Event	Event Frequency	Interface	Initiated By	Event Description
Request Communications Support	One service per request.	<p>Process: DCE Security Server</p> <p>Libraries: EcSelogin, EcSeLogincontext</p> <p>Classes: EcSelogin, EcSeLogincontext</p> <p>Library: EcPf</p> <p>Classes: EcPfManagedServer, EcPfclient</p> <p>Library(Common): EcUr</p> <p>Class: EcUrServerUR</p> <p>Process: Library: Event</p> <p>Class: EcLgErrormsg</p>	<p>Processes: EcDpAtSSAPGui, EcDpAtInsertExeTarFile, EcDpAtInsertStaticFile, EcDpAtInsertTestFile, EcDpAtStageDAP, EcDpAtGetMCF</p> <p>Library: EcDpAtDsrv</p> <p>Classes: DpAtSSAPGui, DpAtInsertTarFile, DpAtInsertStaticFile, DpAtInsertTestFile, DpAtStageDap, DpAtGetMCF</p>	<p>The DCCI CSCI Process Framework provides a library of services available to each SDPS and CSMS process. The services required to perform the specific process assignments are requested by the process from the Process Framework. These services include: DCE support, file transfer services, Network & Distributed File Services, Bulk Data transfer services, file copying services, name/address services, password services, Server Request Framework (SRF), UR, Error/Event logging, message passing, Fault Handling services, User Authentication services, and Mode information.</p>

Table 4.7.2.5-1. AITTL Process Interface Events (2 of 2)

Event	Event Frequency	Interface	Initiated By	Event Description
Initiate Execution	One per tool initialization	UNIX system calls	<i>Process:</i> EcDpAtMgr <i>Library:</i> EcDpAt <i>Class:</i> DpAtMgrLogGuiMain	The EcDpAtMgr initiates the tools and the GUI interface from a menu.
Request Management Services	Per process request.	<i>Process:</i> EcDpAtMgr	<i>Process:</i> EcMsAgSubAgent <i>Library:</i> EcAgInstrm <i>Class:</i> EcAgManager	The EcMsAgSubAgent provides a basic management library of services to the processes, implemented as client or server applications, using the CSS Process Framework. The basic library management of services include: <ul style="list-style-type: none"> • Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training).

Table 4.7.2.5-2. AITTL Process Interface Events (1 of 4)

Event	Event Frequency	Interface	Initiated By	Event Description
Search Advertisements	One the first time each ESDT is used.	<i>Process:</i> EcloAdServer <i>Library:</i> IoAdSearch <i>Class:</i> IoAdApprovedAdvSearchCommand	<i>Processes:</i> EcDpAtInsertTestFile, EcDpAtInsertExeTarFile, EcDpAtInsertStaticFile, EcDpAtSSAPGui <i>Libraries:</i> PICore1 PICore1IF <i>Class:</i> DpAtDsrv, PIDataType	The EcloAdServer receives search requests for subscription event and signature service advertisements from the EcDpAtInsertExeTarFile, EcDpAtInsertStaticFile, and the EcDpAtInsertTestFile processes or the EcDpAtSSAPGui. The same processes or GUI obtains the proper signatures for acquiring data granules from the EcDsScienceDataServer (for the insert and update of metadata within the SDSRV archives).
Request PGE Insert	One per insert request	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsCI <i>Classes:</i> DsCIRequest, DsCICommand	<i>Processes:</i> EcDpAtInsertExeTarFile, EcDpAtInsertStaticFile, EcDpAtInsertTestFile <i>Library:</i> PICore2 <i>Classes:</i> DpAtDsrv, PIResourceRequirement	The EcDpAtInsertExeTarFile, EcDpAtInsertStaticFile, and EcDpAtInsertTestFile send PGE insert requests to the EcDsScienceDataServer for data that defines a PGE and allows it to be scheduled and executed.

Table 4.7.2.5-2. AITTL Process Interface Events (2 of 4)

Event	Event Frequency	Interface	Initiated By	Event Description
Request Data Insert	One per data put into the archive.	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsCI <i>Classes:</i> DsCIRequest, DsCICommand	<i>Processes:</i> EcDpAtInsertTestFile, EcDpAtInsertExeTarFile, EcDpAtInsertStaticFile, EcDpAtSSAPGui <i>Library:</i> DpAtDsrv <i>Class:</i> DpAtDsrv	The EcDpAtInsertStaticFile, EcDpAtInsertTestFile, EcDpAtInsertExeTarFile, and EcDpAtSSAPGui processes send requests to the EcDsScienceDataServer to put various types of data into the archive, from SSAP information to Static files and PGE executables. In response, The EcDpAtInsertExeTarFile, EcDpAtInsertStaticFile, EcDpAtInsertTestFile, and EcDpAtSSAPGui processes get the results of the inserts, including the UR for future acquire requests.
Request MCF	One per MCF request	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsCI <i>Classes:</i> DsCIDescriptor	<i>Process:</i> EcDpAtGetMCF <i>Library:</i> DpAtDsrv <i>Class:</i> DpAtDsrv	The EcDpAtGetMCF process sends a request for a MCF template to the EcDsScienceDataServer. In response, the MCF template is returned and populated.
Request DAPs	One per DAPs request	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsCI <i>Classes:</i> DsCIESDTReferenceCollector DsCIRequest, DsCICommand	<i>Process:</i> EcDpAtAcquireDAP <i>Library:</i> DpAtDsrv <i>Class:</i> DpAtDsrv	The EcDpAtStageDAP requests DAPs from the SDSRV Archives (at the EcDsScienceDataServer) based on the UR. In response, the DAPs are returned and stored on the local AITTL disk.
Request Product	One per user request.	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsDdSSh <i>Classes:</i> DsDdScheduler, DsDdRequestMgrReal	<i>Process:</i> EcDpAtStageDAP <i>Library:</i> DpAtDsrv <i>Class:</i> DpAtDsrv	The EcDpAtStageDAP sends requests to the EcDsScienceDataServer for particular data granules to be pushed, via the FTP service, onto the DPS science processor as input for data processing or for SSIT work.

Table 4.7.2.5-2. AITTL Process Interface Events (3 of 4)

Event	Event Frequency	Interface	Initiated By	Event Description
Insert/Update Granules and PGE information	One per insert/update of granule information	PDPS Database	<i>Processes:</i> EcDpAtInsertStaticFile, EcDpAtInsertExeTarFile <i>Library:</i> PICore1 <i>Classes:</i> DpAtDsrv, PIDataGranule	Insert/update granule information: <ul style="list-style-type: none"> received from a static granule insert request about a modified, existing PGE
Insert/Update PGE Info	One PGE per request	PDPS Database	Sybase Server (COTS)	The Sybase Server inserts or updates PGE information in the PDPS database.
Request PGE Insertion or Update	One per insert/update request	Sybase Server (COTS)	<i>Process:</i> EcDpAtRegisterPGE <i>Library:</i> PICore2 <i>Class:</i> PIResourceRequirement	The EcDpAtRegisterPGE process sends insert or update requests to the Sybase Server to add or modify PGE information in the PDPS database to perform data processing tasks.
Request ODL File	One per ODL file request	PGE/ESDT ODL Files	<i>Processes:</i> EcDpAtCheckODL, EcDpAtRegisterPGE <i>Library:</i> DpAtMetadata <i>Classes:</i> DpAtDatabase, DpAtCheckOdl, DpAtScienceMd	In response to a request for an ODL File, the EcDpAtCheckODL and EcDpAtRegisterPGE processes receive data in "parameter = value" format about a PGE, its inputs and outputs, and scheduling information.

Table 4.7.2.5-2. AITTL Process Interface Events (4 of 4)

Event	Event Frequency	Interface	Initiated By	Event Description
Initiate Execution	One per tool initialization	<i>Processes:</i> EcDpAtInsertExeTarFile, EcDpAtInsertStaticFile, EcDpAtInsertTestFile, EcDpAtStageDAP, EcDpAtGetMCF, EcDpAtMgrLogDump, EcDpAtRegisterPGE, EcDpAtCheckODL, EcDpAtCreateODLTemplate, EcDpAtBadFunc, EcDpAtPrologs, EcDpAtCheckHdfFile, EcDpAtCheckPCF, EcDpAtBinDiffGui, EcDpAtBadFuncGui, EcDpAtOpDbGUI, EcDpAtSSAPGui	<i>Process:</i> EcDpAtMgr <i>Library:</i> EcDpAt <i>Class:</i> DpAtMgrLogGuiMain	The EcDpAtMgr initiates the tools and the GUI interfaces from a menu using UNIX system calls.
Request SSAP Information	One per SSAP information request	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsCI <i>Classes:</i> DsCIRequest, DsCICommand, DsCIESDTRreferenceCollector	<i>Process:</i> EcDpAtSSAPGui <i>Libraries:</i> DpAtSSAP, DpAtDsrv <i>Classes:</i> DpAtSSAPManager, DpAtDsrv	The EcDpAtSSAPGui sends requests to the EcDsScienceDataServer for information about SSAPs, including names of existing SSAPs and the components associated with a specific SSAP.
Insert SSAP Information	One per SSAP	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsCI <i>Classes:</i> DsCIRequest, DsCICommand	<i>Process:</i> EcDpAtSSAPGui <i>Library:</i> DpAtDsrv <i>Class:</i> DpAtSSAPManager, DpAtDsrv	The EcDpAtSSAPGui sends requests to the EcDsScienceDataServer to insert new SSAP or update existing SSAP information.

4.7.2.6 Algorithm and Test Tools Data Stores

Table 4.7.2.6-1 provides descriptions of the data stores shown in the AITTL CSCI architecture diagram.

Table 4.7.2.6-1. AITTL Data Stores

Data Store	Type	Functionality
PGE/ESDT ODL files	Files	These files are written in <i>parameter = value</i> formats to define the inputs and outputs of a PGE and any relevant scheduling information (including Production Rules), and are created by the Instrument Teams and the SSIT personnel.
PDPS database	Database	<p>The PDPS database is replicated at each site for fault handling and recording purposes. The PDPS database holds all the persistent data including:</p> <ul style="list-style-type: none"> • Resource information entered with the Resource Planning utilities • PGE and data type information entered at SSIT • Production Request, Data Processing Request and Data Granule information entered using the Production Request Editor • Plan information entered using the Production Planning Workbench

4.7.3 Data Processing Hardware Components

4.7.3.1 Science Processor Hardware CI (SPRHW) Description

Science Processor hardware (SPRHW) consists of the Science Processor Hardware and the Queuing Server Hardware.

The Science Processor Hardware features Redundant Arrays of Inexpensive Disks (RAID) devices set at RAID Level 3. The Queuing Server Hardware features attached disk packs for additional storage. X-terminals are also provided as part of the SPRHW for additional user access to ECS.

Science Processor

The Science Processor is based on a 64-bit SGI machine. Each Science Processor consists of 12 or 16 processors (See 920-TDx-001 series of base-line documents to see how the configuration is determined). Each Science Processor has one to six GB of memory with eight-way interleaving to improve the input/output (I/O) performance between the processors and memory (see 920-TDx-001 series of base-line documents). The SGI architecture is configured with I/O subsystems attached to the back plane and referred to as PowerChannel2 or IO4 cards. Each IO4 provides serial and parallel connections, two fast-wide differential SCSI-2 channels, and space for two High Input Output (HIO) controller cards. HIO controller card includes a HIPPI card, a FDDI card, and a card to support three SCSI-2 channels.

The number of IO cards specified for each Science Processor is determined by allocating HIO slots to the FDDI and HIPPI interfaces, and counting the number of SCSI-2 interfaces required. The number of internal and external SCSI-2 devices supported by the system determines the required number of SCSI-2 interfaces. The first SCSI-2 channel is delegated to internal devices, i.e., CD-ROMs, floppy disk drives, and tape drives. Internal disks ranging in aggregate size from eight GB to 12 GB are allocated to the second SCSI-2 channel. External disk arrays are allocated

to subsequent SCSI-2 channels; the number of channels is based on the required throughput of the external file systems (see 920-TDx-001 and 922-TDx-015 series of base-line documents).

The internal disks of the Science Processor are only used to provide swap space for the operating system and to provide file system space for the operating system and applications (see 920-TDx-001 and 922-TDx-015 series of base-line documents).

A FDDI sub-network is implemented at each site to support the Planning and Data Processing Subsystems (PDPS). Each processing unit of Science Processor (including the Queuing Server) is dual-attached to the PDPS FDDI sub-network (see 920-TDx-001, 921-TDx-002, 921-TDx-003, and 921-TDx-004 series of base-line documents).

Data transfer requirements between the Science Processor and the DSS are met with a switched HIPPI network implemented via a central HIPPI switch with switched 800 Mbps interface ports. The Science Processor connects directly to the DSS hosts (see 920-TDx-001, 921-TDx-002, 921-TDx-003, and 921-TDx-004 series of base-line documents).

Because the Science Processor does not provide long term, secure storage, data backup and recovery are not provided. The Science Processor storage is used to hold ancillary data files and data granules for short periods of time. If a file system failure occurs within a Science Processor, the algorithms, ancillary data files, and data granules are recoverable from the Data Server.

The dual-ring implementation of FDDI provides fault tolerance. Media failures within the FDDI fabric do not result in any loss of service and does not require re-configuration. Neither is it required to have multiple physical communications paths to each host. Hosts within the Science Processor use dual-attached station cards.

Failure recovery for the HIPPI switch used is supplied by stocking spare Line Replaceable Units of the switch power supplies, interface cards, fan. If an individual interface card fails, a host is re-configured to a hot spare interface card by moving two cables and sending the activating software commands to the switch. If the control module fails, it is replaced with a spare module and the switch is re-configured. In the event of a failure of the entire switch, the switch is either replaced or repaired.

Queuing Server

The Queuing Server with a Sybase database directs AutoSys to load and execute the daily production schedule of a DAAC.

The Queuing Server is based on the SUN Server or the SUN workstation depending on the DAAC site capacity requirements. With a load requirement on the AutoSys and Sybase database for a 24-hour production run of 187,200 jobs, the Queuing Server uses four Ultra-SPARC processors. DAAC sites with smaller production runs are equipped with a Queuing Server based on a dual-processor, SPARC-based workstation (see 920-TDx-001 series of baseline documents).

Each Queuing Server is equipped with a minimum of 384 MB of memory to meet the AutoSys and Sybase database processing requirements (see 920-TDx-001 series of baseline documents).

The internal disks on a Queuing Server are only used to provide swap space for the operating system and to provide file system space for the operating system and applications (see 920-TDx-001 and 922-TDx-014 series of baseline documents).

Additional storage needed to support the Sybase database and to back-up the database from the Planning and Data Processing Subsystems (PDPS) Database Management System Server is via a SCSI-2 interface. To support failure recovery of the Sybase databases, two times the normal operating storage is available (see 920-TDx-001 and 922-TDx-014 series of baseline documents).

A FDDI sub-network is implemented at each site to support the Planning and Data Processing Subsystems (PDPS). Each processing unit of a Science Processor and the Queuing Server is dual-attached to the PDPS FDDI sub-network (see 920-TDx-001, 921-TDx-002, 921-TDx-003, and 921-TDx-004 series of baseline documents).

The AutoSys database located on the Queuing Server is replicated by the MSS Backup Server (See Section 4.9.16: MHWCI Description) to a physical location on the PDPS Database Management System Server. When a disk or database failure occurs on the primary database, AutoSys continues to operate using the backup database on the PDPS Database Management System Server.

The dual-ring implementation of the FDDI provides fault tolerance. Most media failures within the FDDI fabric do not result in any loss of service or require re-configuration of the hardware. Given the fault tolerance of FDDI, it is not required to have multiple physical communications paths to each host in the SPRHW. Each host within the SPRHW uses dual-attached station cards.

4.7.3.2 Algorithm Quality Assurance Hardware CI Description

Algorithm Quality Assurance Hardware (AQAHW) used to validate the quality of ECS products include non-science QA, in-line QA, and SCF-based QA. Non-science QA is specified by the DAAC Operations staff and includes data integrity checks on the data products and the metadata. In-line QA is a form of science QA validating product content using science algorithms. The ECS provides support for SCF-based QA by providing archive and communications capacity for the SCFs to sample and validate the contents of the products.

The AQAHW is an AQA workstation and a Disk/RAID Driver.

AQA Workstation

The AQA workstation provides a software execution environment equivalent to the AI&T software execution environment in order to facilitate the use of the AQA workstation for AI&T when necessary. Also, the AQA supports complex data viewing techniques.

The AQA workstation is a 64-bit SGI machine. For information on the processors used, see the 920-TDx-001 series of base-line documents. The AQA workstation is equipped with a minimum of 128 MB of memory. The AQA workstation is equipped with four EISA slots. These EISA slots have a transfer rate of 33 MB per second. Additionally, the AQA workstation is equipped with two fast SCSI-2 connections. The FDDI interface card and the graphics subsystem use two EISA slots.

The internal disk provides swap space for the operating system and file system space for the operating system and applications (see 920-TDx-001 series of base-line documents). There are no external storage arrays.

A FDDI sub-network is implemented at each site to support the Planning and Data Processing Subsystems (PDPS). The AQAHW uses a single-attached FDDI interface to connect with the remaining members of the PDPS suite (See 920-TDx-001, 921-TDx-002, 921-TDx-003, and 921-TDx-004 series of base-line documents). The AQAHW can also communicate with other ECS hardware items and the external world via the DAAC FDDI switch.

The function of the FDDI is not critical to AQAHW data processing and in the event of a failure, the faulty hardware (including the FDDI interface) is either repaired or replaced by a certified technician.

AQA Disk/RAID Driver

The AQA Disk/RAID Driver supports the AQA Workstation by providing storage for the QA and SCF-based QA activities.

The AQA Disk/RAID Driver is a 64-bit SGI machine. For information on the processors, see the 920-TDx-001 series of base-line documents. The AQA Workstation is equipped with a minimum of 128 MB of memory.

The internal disk provides swap space for the operating system and file system space for the operating system and applications (See 920-TDx-001 and 922-TDx-003 series of base-line documents).

SGI storage units referred to as “Vaults” are attached to the AQA Disk/RAID Driver, via a SCSI-2 interface, to provide the additional storage space (See 920-TDx-001 and 922-TDx-003 series of base-line documents) to support QA.

A FDDI sub-network is implemented at each site to support the Planning and Data Processing Subsystems (PDPS). The AQAHW uses a single-attached FDDI interface to connect with the remaining members of the PDPS suite. The AQAHW also communicates with other ECS units and the external world via the DAAC FDDI switch (See 920-TDx-001, 921-TDx-002, 921-TDx-003, and 921-TDx-004 series of base-line documents). The function of the FDDI is not critical to AQAHW data processing and in the event of a failure, the faulty hardware (including the FDDI interface) is either repaired or replaced by a certified technician.

4.7.3.3 Algorithm Integration and Test Hardware CI Description

The Algorithm Integration and Test Hardware (AITHW) Configuration Item is the hardware to support the system level software validation, integration, and test and the integration and test of science software at a DAAC.

AITHW contains an AIT workstation and an AIT/Sybase Server with a laser printer and X-terminals to provide additional user access.

AIT Workstation

The AIT workstation is a 64-bit SUN workstation class machine with 128 MB of memory (See 920-TDx-001 series of base-line documents). The AIT workstation is for building and testing software in the AIT environment.

The AIT/Sybase Server internal disk provides swap space for the operating system and file system space for the operating system and applications (See 920-TDx-001 series of base-line documents). An external disk pack is attached via the SCSI port to provide additional storage.

A FDDI sub-network is implemented at each site to support the Planning and Data Processing Subsystems (PDPS). The AITHW uses a single-attached FDDI interface to connect with the remaining members of the PDPS suite. The AITHW also communicates with other ECS units and the external world via the DAAC FDDI switch (See 920-TDx-001, 921-TDx-002, 921-TDx-003, and 921-TDx-004 series of base-line documents).

The function of this component is not critical to data processing. In the event of a component failure, the faulty component (including the FDDI interface) is either replaced or repaired by a certified SUN technician.

AIT/Sybase Server

The AIT/Sybase Server is the hardware tools and database support for AIT.

The AIT/Sybase Server is a 64-bit SUN workstation class machine. For information on the processor utilized in this workstation see the 920-TDx-001 series of base-line documents. The AIT/Sybase Server is equipped with a minimum of 256 MB of memory (See 920-TDx-001 series of base-line documents).

The AIT internal disk provides swap space for the operating system and file system space for the operating system and applications (See 920-TDx-001 series of base-line documents) with external multi-packs attached via the SCSI port to provide additional storage.

A FDDI sub-network is implemented at each site to support the PDPS. The AITHW use either a single or dual-attached FDDI interface to connect with the other members of the PDPS suite. The AITHW communicates with other ECS units and the external world via the DAAC FDDI switch (See 920-TDx-001, 921-TDx-002, 921-TDx-003, and 921-TDx-004 series of base-line documents).

The function of the FDDI is not critical to AITHW data processing. In the event of a failure, the faulty hardware is either repaired or replaced by a certified technician.

The dual-ring implementation of FDDI provides a fault tolerant capability. Media failures within the FDDI fabric do not result in the loss of service and do not require hardware re-configuration. With the inherent fault tolerance of FDDI, multiple physical communications paths to each host are not necessary.

4.8 Communications Subsystem Overview

The Communications Subsystem (CSS) provides the capability to:

- Transfer information internal to the Earth Observing System Data and Information System (EOSDIS) Core System (ECS)
- Transfer information between the ECS sites
- Provide connections between the ECS users and service providers
- Manage the ECS communications functions
- Provide services requested to support System Management Subsystem (MSS) operations
- Retrieve attribute-value pairs from the Configuration Registry

Communications Subsystem Context Diagram

Figure 4.8-1 is the Communications Subsystem (CSS) context diagram and Table 4.8-1 provides descriptions of the interface events shown in the CSS context diagram. **NOTE:** In Table 4.8-1, Request Communications Support is shown as a single event to simplify the table and provide a list of services available from CSS to the other SDPS and CSMS subsystems.

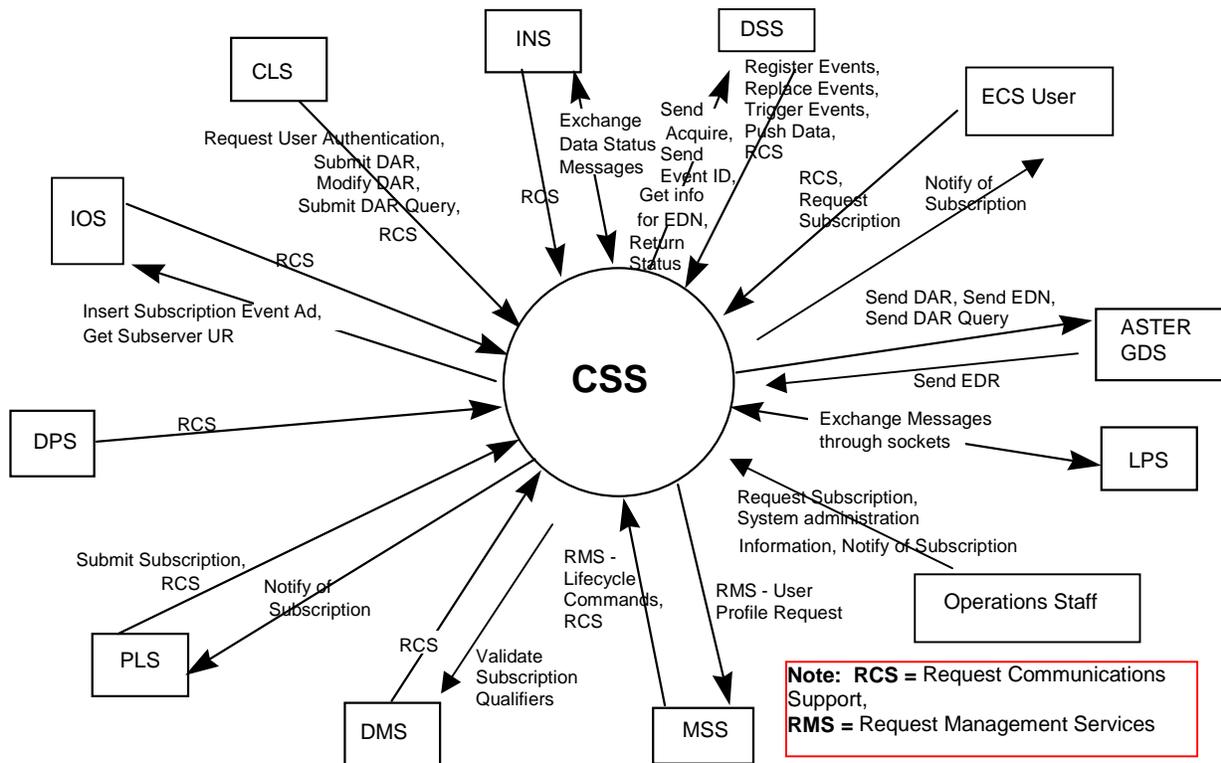


Figure 4.8-1. Communications Subsystem (CSS) Context Diagram

Table 4.8-1. Communications Subsystem (CSS) Interface Events (1 of 2)

Event	Interface Event Description
Send Acquire	An "acquire" (instruction to obtain data) is created by the CSS and sent to the DSS via remote procedure call. This is similar to the "Request Product" interface event, except it applies to EDOS expedited data. Also, the Subscription Server sends an "acquire" command to the DSS when an "acquire" action is specified in a subscription.
Send Event ID	The CSS sends Event IDs to the DSS when ESDTs are installed or when ESDTs are updated by adding additional events.
Get info for EDN	Expedited Data Set Notification (EDN) information is obtained from the DSS, by request, and used by the CSS to send messages to users at the ASTER GDS.
Return Status	Status returned by the CSS to the DSS to simply indicate that the request was received, not that the action succeeded.
Register Events	The DSS sends the subscription events for an Earth Science Data Type to the CSS Subscription Server when an ESDT is installed into the system or when an ESDT is updated by adding additional events.
Replace Events	The DSS sends the updated subscription events with modified qualifiers for an Earth Science Data Type (ESDT) to the CSS Subscription Server when an ESDT is updated. This event replaces the original event in the DSS.
Trigger Events	The DSS notifies the CSS (via an event trigger) when a subscription event occurs on an ESDT Service.
Push Data	The DSS assembles instructions to send data to the ASTER GDS or other external users via the CSS. The DSS pushes data, via the FTP service and followed by a signal file, to the destination specified in an acquire instruction (by particular ESDTs that function this way).
Request Communications Support (RCS)	The CSS provides a library of services available to each SDPS and CSMS subsystem. The services required to perform the specific subsystem assignments are requested by the subsystem from the CSS. These services include: DCE support, file transfer servers, Network and Distributed File Servers, Bulk Data transfer services, file copying services, name/address services, password services, Server Request Framework (SRF), UR, Error/Event logging, message passing, Fault Handling services, User Authentication services, Mode Information and retrieving the requested configuration attribute-value pairs from the Configuration Registry for ECS applications that request them.
Request Subscription	The Operations Staff or an ECS User submits a request for notification of a specific event occurring within the system to the CSS Subscription Server. For example: subscribing to the insert of a particular granule type through the CLS.
Notify of Subscription	In response to a subscription request, a message (containing the UR of the granule inserted into the DSS) is sent to the PLS Subscription manager or the On-Demand Processing Manager, via the CSS, and sent to the Operations staff or to ECS users.
Send DAR	The CSS sends the DAR to the ASTER GDS Storage Server.
Send EDN	The CSS stores the EDN messages with URs, time range, etc., and sends the EDN to the MSS to forward to the ASTER GDS.
Send DAR Query	The CSS sends the DAR query to the ASTER GDS DAR Server.

Table 4.8-1. Communications Subsystem (CSS) Interface Events (2 of 2)

Event	Interface Event Description
Send EDR	The ASTER GDS personnel select the EDN as needed and send an EDR to the CSS to forward to the E-mail Parser Gateway.
Exchange Messages through sockets	The CSS sends and receives data status messages from the LPS through sockets.
System administration information	The Operations staff requests and receives information on system administration including application administration, fault metrics, performance metrics and system alarms.
Request Management Services	<p>The MSS provides a basic management library of services to the subsystems, implemented as client or server applications, using the CSS Process Framework. The basic management library of services includes:</p> <ul style="list-style-type: none"> • Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training). • User Profile Request – The MSS provides requesting subsystems with User Profile parameters such as e-mail address and shipping address to support their processing activities.
Validate Subscription Qualifiers	The Data Dictionary is queried for type and range information to validate qualifiers.
Submit Subscription	The PLS creates a subscription, sent to the CSS, using the advertisement for subscribing to an insert event for an ESDT. In response, PLS receives a corresponding subscription identifier.
Insert Subscription Event Ad	The IOS receives requests to insert subscription event service advertisements from the CSS Subscription Server.
Get Subserver UR	The CSS retrieves the correct subscription server UR from the IOS.
Request User Authentication	The WKBCH CSCI submits a request for user authentication to the DCCI CSCI (DCE CSC).
Submit DAR	The CLS sends the parameters required for submittal of Data Acquisition Requests for ASTER instrument data to the CSS. In response, a DAR Identifier is sent back to the CLS.
Modify DAR	The CLS sends parameters to modify an existing Data Acquisition Request for ASTER instrument data collection to the CSS. A status value is returned to the CLS.
Submit DAR Query	The CLS sends the parameters required for querying DARs to the CSS as one of the following three queries: queryxARContents, queryxARScenes, or queryxARSummary. The results of the query are returned to the CLS.
Exchange Data Status Messages	Data status messages are sent to and from the CSS Gateways via a Remote Procedure Call (RPC). A Data Availability Notice (DAN) is sent to the INS and additional data status messages are exchanged with the INS.

Communications Subsystem Structure

Note: The CSS logical names used in this document do not exactly match the physical names in the directory structure where the software is maintained. Therefore, after the logical name of each Computer Software Component (CSC) in parentheses, there is a physical directory structure name where the software is found. For example, the DCCI CSCI software can be found under the directory structure Distributed Object Framework (DOF) and the Server Request Framework software can be found under the directory structure /ecs/formal/common/CSCI_SRF.

The CSS is composed of one CSCI, the Distributed Computing Configuration Item (DCCI, the software is found in directory DOF) and one HWCI. The CSS software is used to provide communication functions, processing capability, and storage.

Use of COTS in the Communications Subsystem

- RogueWave's Tools.h++

The Tools.h++ class libraries provide basic functions and objects such as strings and collections. These class libraries must be installed with the CSS software to enable the CSS processes to run.

- RogueWave's DBTools.h++

The DBTools.h++ C++ class libraries provide interaction, in an object-oriented manner, to the Sybase database SQL server. The DBTools provide a buffer between the CSS processes and the relational database used. These class libraries must be installed with the CSS software to enable the Subscription Server to run and enable the clients to perform queries of subscription server database information.

- RogueWave's Net.h++

The Net.h++ C++ class libraries provide functions and templates that facilitate writing applications, which communicate with other applications. These class libraries must be installed with the CSS software to enable the Landsat 7 Gateway to run.

- ICS' Builder Xcessory

The Builder Xcessory GUI builder tool provides the capability to modify the displays of the Subscription Server Operator GUI. The tool also generates the C++ code producing the Operator GUI display at run time. There is no operational part of Builder Xcessory used at CSS run-time.

- Sybase Server

The Sybase SQL server provides access for the Subscription Server to insert, update, and delete Subscription Server database information. The Sybase SQL Server must be running during CSS operations for the Subscription Server to execute database requests.

In addition, the Configuration Registry stores configuration values for ECS applications in the Sybase database. The Configuration Registry Server retrieves the values from the database via a Sybase SQL server.

- Distributed Computing Environment

The Distributed Computing Environment (DCE) provides a basis for building manageable, secure, distributed, interoperable, and portable applications across heterogeneous platforms. DCE offers APIs for application developers and includes commands sets for administrator and user application generation. DCE provisions include security, distributed file, cell directory, distributed time, and thread services. Using Remote Procedure Calls (RPCs), a feature of DCE, resources and files on a distributed network can be accessed.

- UNIX Network Services

UNIX Network Services contain DNS, NFS, E-mail service, FTP, and TCP/IP capabilities.

4.8.1 The Distributed Computing Configuration Item Software Description

The DCCI CSCI (the software is found in directory DOF) consists mainly of COTS software and hardware providing servers, gateways, and software library services to other SDPS and CSMS CSCIs. The CSCI is composed of 17 computer software components (CSCs) briefly described here followed by a description of the HWCI.

The CSCI is composed of 17 computer software components (CSCs) briefly described here as processes followed by a description of the HWCI.

1. The Subscription Server (SBSRV, the software is found in directory SUBSCRIPTION) Provides the capability to register advertised events, accept subscriptions, and process subscriptions upon event notification. Events are made available to users through the advertisement service during the user registration process. Subscriptions are submitted against an advertised event. The subscription can be qualified by metadata attribute values and can also include information specifying a particular action to be performed on behalf of the subscriber. Upon event notification, all subscriptions for the event and any associated actions are performed. Event examples are science granule insertion, metadata update, new advertisement, and new schema exports to DDICT. Additionally, subscribers receive notification an event was triggered, either via e-mail or through inter-process communication.
2. The ASTER DAR Gateway Server (the software is found in directory RELB_GATEWAY/DAR) provides interoperability between the CSS Message Oriented middleware of JEST (Java Earth Science Tool) Objects (MOJO) Gateway and the DAR API with an interface to the ASTER Ground Data System (GDS) servers. DAR communications are part of the ECS and ASTER GDS interface. The ASTER GDS provides the ground support for mission operations and science data processing for the ASTER instrument aboard the TERRA spacecraft. The DAR Server is located in Japan and transparently interacts with the ASTER Operations Segment (AOS) xAR Server and xAR data base at the back end to provide a data acquisition service to its clients. The DAR Server allows data base access to ECS Clients via an API. DAR related communications between ECS and the ASTER GDS are accomplished through the ASTER GDS provided APIs. The ASTER GDS provided APIs are integrated into the

DAR Communications Gateway. The DAR Communications Gateway Server is hosted at the EROS Data Center (EDC).

3. The ASTER E-Mail Parser Gateway Server (the software is found in directory RELB_GATEWAY/EmailParser) supports the automated delivery of ASTER Expedited Data Sets (EDS) from the ECS to the ASTER GDS. EDS are defined as raw satellite telemetry data processed in time-ordered instrument packets. The packets are separated into files for a given down link contact. The E-mail Parser forwards notification (an Expedited Data Set Notification or EDN) to the ASTER GDS when EDS are received. The E-mail Parser receives requests (an Expedited Data Set Request or EDR) from the ASTER GDS to deliver EDS. The ECS provides EDS to the ASTER GDS for evaluating the operation of the instrument. Level 0 EDS produced at the DAAC are staged for up to 48 hours for delivery to Science Computing Facilities investigators. Subscription notifications are sent to the E-mail Parser. The E-mail Parser properly formats the EDN mail messages and sends them to the ASTER GDS. The MSS ASTER E-mail Header Handler attaches an ICD approved header. If the ASTER GDS decides to order the EDS from ECS, it sends an e-mail to the E-mail Parser. The E-mail Parser creates and submits the corresponding acquire request to the Science Data Server. The Science Data Server requests a distribution of the EDS via the ECS Distribution Server to the ASTER GDS.
4. The Landsat 7 Gateway Server (the software is found in directory RELB_GATEWAY/LANDSAT_GATEWAY) is required to facilitate communications between the ECS and the Landsat 7 Processing System (LPS) since the LPS is not DCE compatible. The Landsat 7 Gateway Server provides ECS users access to data collected by the Enhanced Thematic Mapper Plus (ETM+) instrument on the Landsat 7 satellite. The Landsat 7 project processes the raw instrument data into Level 0R data. The Landsat 7 project provides the Level 0R data to the ECS for ingest, archive and distribution. All ECS registered users are permitted access to Landsat 7 Level 0R data; metadata and browse data archived by the ECS.
5. The MOJO Gateway Server (the software is found under RELB_GATEWAY/Mojo) provides a common interface and network address for the following distributed ECS services: User profiles, Advertisements, Subscriptions, and DAR submittals, modifications, and queries. All services are accessible from the Java front end. The MOJO Gateway Server routes all DAR requests to the ASTER DAR Gateway Server, which sends them to the ASTER GDS via APIs, provided by the ASTER GDS. The MOJO Gateway Server sends requests to retrieve user profiles from the MSS Registration User Server. Advertisements are retrieved from the IOS Advertising Server, and subscriptions are placed with the CSS Subscription Server.
6. The Configuration Registry Server (the software is found in directory REGISTRY) provides a single interface to retrieve configuration attribute-value pairs for ECS Applications from the Configuration Registry Database. Configurable run-time parameters for Process Framework-based ECS Applications (including clients and servers) are stored in the Configuration Registry Database. Upon startup, the Process Framework retrieves this information from the Configuration Registry Server for the application.

7. The DCE service group is a COTS software set of Name, Security and Time Services.
 - The Cell Directory Service (the software is found in directory NS for Name Service) provides a link between clients and the ECS servers they need to communicate with to obtain ECS data and services. Servers register their location information in the Cell Directory, independent of physical location. The clients use the Cell Directory to find servers based on an operating mode. This is the primary way clients locate servers and the only way servers advertise their services.
 - The Security Service (the software is found in directory SEC) provides a means for server processes to obtain a valid DCE login context. The Security Service provides secure transfer of data on local and wide area networks. The Security Service provides authentication of users who try to access ECS data or services.
 - The Time Service (the software is found in directory TIME) keeps the ECS computer network system clocks synchronized by monitoring and adjusting the operating system clock for each individual host machine in the network. The Time service provides an API to obtain time in various formats. Some applications need to simulate the current time by applying a delta to the current time. The Time Service retrieves time deltas and applies them to the system time.

The remote file access group provides the capability to transfer and manage files using the following five functions: FTP, FTP Notification, Bulk Data Server (BDS), Network File System (NFS), and Filecopy.

8. FTP (the software is found in directory FTP) is an Internet standard application for file transfers. It is a client/server model in which the FTP is a client program started by the user while the FTP daemon is the server running on the target host. FTP enables a user to retrieve one or more files from a remote server and to send one or more files to a remote server. FTP also provides an insecure password protection scheme for authentication. The FTP application is used to ingest data into the ECS from remote locations and to distribute data to remote servers for users. The INGEST CSCI uses the FTP application to ingest data from external data providers. The STMG T CSCI uses the FTP application to distribute data to users.
9. FTP Notification (the software is found in directory FTP) provides successful completion notifications for FTP (get) data pulls and (put) data pushes. The INGEST CSCI and DDIST CSCI provide notifications to external data providers of data ingest into the ECS or data distribution from the ECS, respectively.
10. BDS (no physical directory) is a non-standard extension to the Network File System (NFS) that handles large file transfers. BDS is a fast file transfer utility to move large data files over high-speed networks such as the High Performance Parallel Interface (HIPPI) communications lines. BDS exploits the data access speed of the NFS file system and data transfer rates of network media, such as HIPPI, to accelerate standard NFS performance. The BDS protocol, XBDS, modifies NFS functions to reduce the time needed to transfer files of 100 megabytes or larger over a network connection. Hosts must be connected to a high-speed network running the Transmission Control Protocol/Internet Protocol suite (TCP/IP). The STMG T CSCI uses BDS to archive data produced by the

Data Processing Subsystem or to distribute ECS science data to the ASTER GDS via the CSS DAR Gateway Server.

11. The NFS (no physical directory) provides a distributed file sharing system among computers. NFS consists of a number of components, including a mounting protocol and server, a file locking protocol and server, and daemons that coordinate basic file service. A server exports (or shares) a filesystem when it makes the filesystem available for use by other machines in the network. An NFS client must explicitly mount a filesystem before using it.
12. The Filecopy utility (the software is found in directory /ecs/formal/common/CSCI_Util/src/copyprog) copies files from a specified source location to a specified destination location with options available for data compression. The STMGT and SDSRV CSCIs use the Filecopy utility to transfer large files. Files are copied from the INGEST staging disks to the SDSRV archives and from the SDSRV archives to the Read-Only Cache. In the SDSRV CSCI, the Filecopy utility is used to copy MCFs to the DDIST CSCI staging disks.
13. The mail support group provides electronic mail service.
 - E-mail (the software is found in directory Email) is a standard Internet feature for asynchronous data transfers. The CSS E-mail service provides an interactive interface and an object-oriented application program interface (API) to send E-mail messages. E-mail messages are sent among ECS users in the United States and between the ECS and the ASTER GDS in Japan. The e-mail sent to the ASTER GDS is sent via the E-mail Parser Gateway and the MSS Aster E-mail Header Handler (it adds an ICD approved header to the message). Messages sent from the ASTER GDS are received by the E-mail utility, passed to the MSS E-mail Header Remover (the ICD approved header is removed) and routed to the appropriate ECS users via the E-mail server.
14. Virtual Terminal (no physical directory) provides the capability for the Operations staff on an ECS platform to remotely log onto another ECS machine.
15. Cryptographic Management Interface (CMI, the software is found in directory AUTHN) provides processes a means for obtaining random passwords and gaining access to non-DCE services like Sybase.
16. The Domain Name Service (DNS, the software is found in directory DN) provides host names and addresses to a specified network by querying and answering queries. DNS provides naming services between the hosts on the local administrative domain and also across domain boundaries. DNS is distributed among a set of servers (name servers); each of which implements DNS by running a daemon called in.named. On the client side, the service is provided through the resolver, which is not a daemon. The resolver resolves user queries by needing the address of at least one name server (provided in a configuration file parameter). Each domain must have at least two kinds of DNS servers (a primary and secondary server) maintaining the data corresponding to the domain. The primary server obtains the master copy of the data from disk when it starts up the in.named. The primary server delegates authority to other servers in or outside of the domain. The secondary server maintains a copy of the data for the domain. When the

secondary server starts in.named, the server requests all data for the given domain from the primary server. The secondary server checks periodically with the primary server for updates. DNS namespace has a hierarchical organization consisting of nested domains like directories. The DNS namespace consists of a tree of domains. See figure 4.8.6.16.1-1 for an illustration of the domain tree hierarchy.

17. The Infrastructure Library provides a set of services including the following.

- Process Framework (PF): The PF is a software library of services, which provides a flexible mechanism (encapsulation) for the ECS Client and Server applications to transparently include specific ECS infrastructure features from the library of services. (Library services include: process configuration and initialization, mode management and event handling, life cycle services (server start-up and shut-down), communications services (message passing, FTP, underlying transport protocol, number of simultaneous threads), naming and directory services (OODCE naming), and set-up of security parameters.) The PF process is the encapsulation of an object with ECS infrastructure features and therefore the encapsulated object is fully equipped with the attributes needed to perform the activities assigned to it. The PF was developed for the ECS custom developed applications and is not meant for use by any COTS software applications. The PF ensures design and implementation consistency between the ECS Client and Server applications through encapsulation of the implementation details of the ECS infrastructure services. Encapsulation therefore removes, for example, the task of each programmer repeatedly writing common initialization code. The PF is built by first developing a process classification for the ECS project from the client/server perspective. Then the required capabilities are allocated for each respective process level and type. PF-based ECS applications use Process Framework to read in their configuration information at startup. PF-based servers use Process Framework to initialize themselves as a DCE server and put it in a listen state to begin to accept requests from appropriate clients.
- Server Request Framework (SRF, the software is found in directory ecs/formal/common/CSCI_SRF): The SRF infrastructure provides the standard for ECS synchronous and asynchronous communications between ECS applications. SRF is used to provide the client-server communications between the INGEST Request Manager and Granule Server, between the MOJO Gateway and the ASTER DAR Gateway, and between clients of the Subscription Server, such as PLS Subscription Manager and the Subscription GUI, and the Subscription Server itself. SRF provides enhanced OODCE RPC, message passing and persistent storage as a CSS support capability with the described features available by subsystem request. SRF uses OODCE RPC calls, which are TCP/IP based.
- Message Passing (the software is found in directory MP-OODCE_N01): Message Passing provides peer-to-peer asynchronous communications service notifying clients of specific event triggers. Provided by subsystem request from the CSS. Message Passing is used in ECS for communication between the Subscription Server and the PLS Subscription Manager in lieu of the more common e-mail that is sent to ECS users as notification of a triggered event. It is an alternative means of communication.

- Universal References (the software is found in directory /ecs/formal/common/CSCI_UR): An UR Provider object from C++ objects generates Universal References (URs) during their run time in virtual memory. The UR is a representation of the original object. URs can be transformed from an object to an ASCII representation and again returned to an object. URs are objects the users and applications use with full capabilities. Once the UR is obtained, the original object can be discarded and later reconstituted and used. URs can refer to objects local or remote to an address space. Therefore, the object does not have to remain in memory, and can, as appropriate, be written to a secondary storage system like a database.
- Event Logging (the software is found in directory Logging): Event logging is the capability of recording events into files and provides a convenient way to generate and report detailed events. All ECS CSCIs use event and error logging as an audit trail for all transactions that occur during the ECS data processing and distributing.
- Server Locator (the software is found in directory NS): The Server Locator is a class that enables servers to register their location without referring to its physical location and be uniquely identified and located in the ECS. Client applications use the Server Locator to find any registered server. The Server Locator is used in ECS in any client-server DCE-based communication.
- Failure Recovery Framework (the software is found in directory FH): The Failure Recovery Framework provides a general purpose fault recovery routine enabling client applications to reconnect with servers after the initial connection is lost. This is accomplished through the DCE Cell Directory Service (CDS), through which the Failure Recovery Framework can determine whether a server is listening. The Failure Recovery Framework provides a default and configurable amount of retries and duration between retries. This fault recovery takes effect for each attempt by the client to communicate with the server for all applications that employ the Failure Recovery Framework.
- EcPo Connections (the software is found in directory /ecs/formal/common/CSCI_DBWrapper): A suite of classes providing a basic set of database connection management methods and an error handling mechanism for database users, which is found in the DBWrapper directory of the Infrastructure Library Group.
- CSS software is executed on multiple hardware hosts throughout the ECS system to provide communication functions, processing capability, and storage. The software and hardware relationships are discussed in the CSS Hardware CI description.

4.8.2 The Distributed Computing Configuration Item Context

Figure 4.8.2-1 is the Distributed Computing Configuration Item (DCCI) CSCI context diagram. The diagram shows the events sent to the DCCI CSCI and the events the DCCI CSCI sends to other CSCIs. Table 4.8.2-1 provides descriptions of the interface events shown in the DCCI CSCI context diagram.

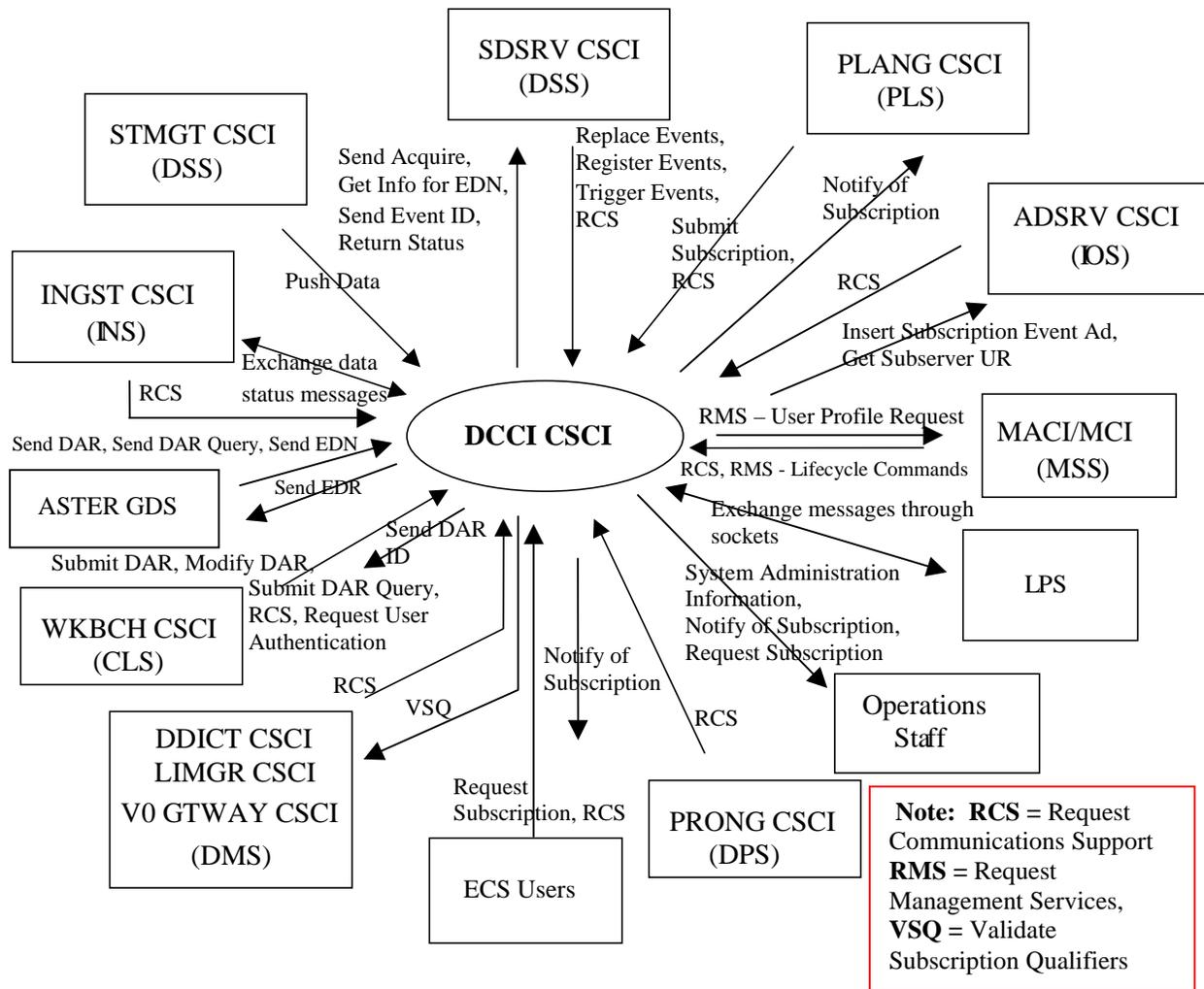


Figure 4.8.2-1. Distributed Computing Configuration Item (DCCI) CSCI Context Diagram

Table 4.8.2-1. Distributed Computing Configuration Item (DCCI) CSCI Interface Events (1 of 3)

Event	Interface Event Description
Replace Events	The SDSRV CSCI sends the updated subscription events with modified qualifiers for an Earth Science Data Type (ESDT) to the DCCI CSCI (Subscription Server) when an ESDT is updated. This event replaces the original event in the DCCI CSCI.
Register Events	The SDSRV CSCI sends the subscription events for an Earth Science Data Type (ESDT) to the DCCI CSCI when an ESDT is installed into the system or when an ESDT is updated by adding additional events.
Trigger Events	The SDSRV CSCI notifies the DCCI CSCI (via an event trigger) when a subscription event occurs on an Earth Science Data Type Service.
Request Communications Support (RCS)	The DCCI CSCI provides a library of services available to each SDPS or CSMS CSCI. The services required performing the specific CSCI assignments are requested by the CSCI from the DCCI CSCI. These services include: Distributed Computing Environment (DCE) support, file transfer services, Network & Distributed File Services, Bulk Data transfer services, file copying services, name/address services, password services, Server Request Framework (SRF), Universal Reference (UR), Error/Event logging, message passing, Fault Handling services, User Authentication services, Mode Information, and retrieving the requested configuration attribute-value pairs from the Configuration Registry for ECS applications that request them.
Submit Subscription	The PLANG CSCI creates a subscription using the advertisement for subscribing to an ESDT insert event and sends the subscription event to the DCCI CSCI. In response, the PLANG CSCI receives a subscription identifier from the DCCI CSCI.
Notify of Subscription	The DCCI CSCI sends notification to the PLANG CSCI, the operations staff, and users of the occurrence of a subscription event.
Insert Subscription Event Ad	The ADSRV CSCI receives requests to insert subscription event service advertisements from the DCCI CSCI (SBSRV CSC).
Get Subserver UR	The DCCI CSCI (MOJO Gateway Server CSC) retrieves the correct subscription server UR from the ADSRV CSCI.
Request Management Services	<p>The MACI and MCI provide a basic management library of services to the CSCIs, implemented as client or server applications, using the DCCI CSCI Process Framework. The basic management library of services includes:</p> <ul style="list-style-type: none"> • Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training). • User Profile Request – The MCI provides requesting CSCIs with User Profile parameters such as e-mail address and shipping address to support their processing activities.

Table 4.8.2-1. Distributed Computing Configuration Item (DCCI) CSCI Interface Events (2 of 3)

Event	Interface Event Description
Exchange Messages through Sockets	The DCCI CSCI (Landsat 7 Gateway Server CSC) sends and receives data status messages from the LPS through sockets.
System Administration Information	The Operations staff requests and receives information on system administration including application administration, fault metrics, performance metrics and system alarms from the DCCI CSCI.
Request Subscription	An ECS user or Operations Staff member submits a request for a subscription to the DCCI CSCI. The subscription notifies the user or Operations Staff member whenever the desired event occurs in the system.
Validate Subscription Qualifiers	The DCCI CSCI sends queries to the DDICT CSCI for type and range information to validate qualifiers.
Submit DAR	The WKBCH CSCI sends the Data Acquisition Request (DAR) parameters to the DCCI CSCI to submit a DAR to the ASTER GDS.
Modify DAR	The WKBCH CSCI sends the modified DAR parameters to the DCCI CSCI to submit a DAR to the ASTER GDS.
Submit DAR Query	The WKBCH CSCI sends the parameters required for querying DARs to the DCCI CSCI as one of the following three queries: queryxARContents, queryxARScenes, or queryxARSummary. The results of the query are returned to the WKBCH CSCI.
Request User Authentication	The WKBCH CSCI submits a request for user authentication to the DCCI CSCI (DCE CSC).
Send DAR ID	The ASTER DAR Gateway Server extracts the returned DAR ID and sends it to the Java DAR Tool, via the MOJO Gateway Server.
Send DAR	The DCCI CSCI (ASTER DAR Gateway Server) sends the DAR to the ASTER GDS Storage Server.
Send DAR Query	The DCCI CSCI sends the DAR query to the ASTER GDS DAR Server.
Send EDN	The DCCI CSCI (E-mail Parser Gateway Server CSC) stores the EDN messages with URs, time range, etc., and sends the EDN to the MSS to forward to the ASTER GDS.
Send EDR	The ASTER GDS personnel select the EDN as needed and send an EDR to the MCI to forward to the DCCI CSCI (E-mail Parser Gateway Server CSC).
Exchange Data Status Messages	Data status messages are sent to and from the DCCI CSCI via Remote Procedure Calls (RPCs). A Data Availability Notice (DAN) is sent to the INGST CSCI and afterwards additional data status messages are exchanged with the INGST CSCI.
Push Data	The STMGT CSCI pushes data (i.e., EDS), via the FTP service, to the DCCI CSCI for data distribution per user request. A signal file is also sent to indicate the completion of the file transfer for some ESDTs.
Send Acquire	An "acquire" (instruction to obtain data) is created by the DCCI CSCI and sent to the SDSRV CSCI via remote procedure call. This is similar to the "Request Product" interface event, except it applies to EDOS expedited data.

Table 4.8.2-1. Distributed Computing Configuration Item (DCCI) CSCI Interface Events (3 of 3)

Event	Interface Event Description
Get Info for EDN	Expedited Data Set Notification (EDN) information is obtained from the SDSRV CSCI, by request, and used by the DCCI CSCI to send messages to users at the ASTER GDS.
Send Event ID	The DCCI CSCI sends Event IDs to the SDSRV CSCI when ESDTs are installed or when ESDTs are updated by adding additional events.
Return Status	Status returned by the DCCI CSCI to the SDSRV CSCI to simply indicate that the request was received, not that the action succeeded

4.8.3 Distributed Computing Configuration Item Architecture

An architecture diagram is not applicable for the DCCI CSCI. However, table 4.8.3-1 shows the mapping between SDPS/CSMS CSCIs and CSS CSCs.

Table 4.8.3-1. SDPS/CSMS CSCI to CSS CSC Mappings (1 of 3)

SDPS/CSMS CSCI	CSS CSC	Process Used	CSS Libraries Used
SDSRV	<ul style="list-style-type: none"> - Distributed Computing Environment (DCE) - Subscription Server (SBSRV) - E-mail Parser Gateway Server 	<ul style="list-style-type: none"> - DCE Security Server - EcSbSubServer - EcCsEmailParser 	<ul style="list-style-type: none"> - Process Framework (PF) - ServerUR - Error Logging - Event Logging - Universal Reference (UR)
STMGT	<ul style="list-style-type: none"> - DCE - File Transfer Protocol (FTP) - Network File System (NFS) - Filecopy 	<ul style="list-style-type: none"> - DCE Security Server - FTP_POPEN - NFS Client - EcUtFileCopy 	<ul style="list-style-type: none"> - PF - ServerUR - Error Logging - Event Logging - UR
INGST	<ul style="list-style-type: none"> - DCE - E-Mail Parser Gateway Server - FTP - NFS 	<ul style="list-style-type: none"> - DCE Security Server - EcCsEmailParser - FTP_POPEN - NFS Client 	<ul style="list-style-type: none"> - PF - ServerUR - Error Logging - Event Logging - UR - Fault Handling Services - Server Request Framework (SRF)
EOS Data Gateway	<ul style="list-style-type: none"> - DCE 	<ul style="list-style-type: none"> - DCE Security Server 	<ul style="list-style-type: none"> - PF - ServerUR - UR - Error Logging - Event Logging

Table 4.8.3-1. SDPS/CSMS CSCI to CSS CSC Mappings (2 of 3)

SDPS/CSMS CSCI	CSS CSC	Process Used	CSS Libraries Used
WKBCH	<ul style="list-style-type: none"> - DCE - MOJO Gateway Server 	<ul style="list-style-type: none"> - DCE Security Server - EcCsMojoGateway 	<ul style="list-style-type: none"> - PF - ServerUR - Error Logging - Event Logging - UR
ODFRM	N/A	N/A	<ul style="list-style-type: none"> - PF - ServerUR - Error Logging - Event Logging - UR
Desktop	N/A	N/A	<ul style="list-style-type: none"> - PF - ServerUR - Error Logging - Event Logging - UR
DDICT	<ul style="list-style-type: none"> - DCE - SBSRV 	<ul style="list-style-type: none"> - DCE Security Server - EcSbSubServer 	<ul style="list-style-type: none"> - PF - ServerUR - Error Logging - Event Logging - UR
LIMGR	<ul style="list-style-type: none"> - DCE 	<ul style="list-style-type: none"> - DCE Security Server 	<ul style="list-style-type: none"> - PF - ServerUR - Error Logging - Event Logging - UR
V0 Gateway	<ul style="list-style-type: none"> - DCE 	<ul style="list-style-type: none"> - DCE Security Server 	<ul style="list-style-type: none"> - PF - ServerUR - Error Logging - Event Logging - UR
ASTER Gateway	<ul style="list-style-type: none"> - DCE 	<ul style="list-style-type: none"> - DCE Security Server 	<ul style="list-style-type: none"> - PF - ServerUR - Error Logging - Event Logging - UR
ADSRV	<ul style="list-style-type: none"> - DCE - SBSRV - MOJO Gateway Server 	<ul style="list-style-type: none"> - DCE Security Server - EcSbSubServer - EcCsMojoGateway 	<ul style="list-style-type: none"> - PF - ServerUR - Error Logging - Event Logging - UR

Table 4.8.3-1. SDPS/CSMS CSCI to CSS CSC Mappings (3 of 3)

SDPS/CSMS CSCI	CSS CSC	Process Used	CSS Libraries Used
PLANG	<ul style="list-style-type: none"> - DCE - SBSRV - Cryptographic Management Interface 	<ul style="list-style-type: none"> - DCE Security Server - EcSbSubServer - CMI 	<ul style="list-style-type: none"> - PF - ServerUR - Message Passing - Error Logging - Event Logging - UR
PRONG	<ul style="list-style-type: none"> - DCE 	<ul style="list-style-type: none"> - DCE Security Server 	<ul style="list-style-type: none"> - PF - ServerUR - Error Logging - Event Logging
AITTL	<ul style="list-style-type: none"> - DCE 	<ul style="list-style-type: none"> - DCE Security Server 	<ul style="list-style-type: none"> - PF - ServerUR - Error Logging - Event Logging
MCI (CSMS)	<ul style="list-style-type: none"> - E-Mail Parser Gateway Server - SBSRV - ASTER DAR Gateway Server - MOJO Gateway Server 	<ul style="list-style-type: none"> - EcCsEmailParser - EcSbSubServer - EcGwDARServer - EcCsMojoGateway 	<ul style="list-style-type: none"> - PF - ServerUR - Error Logging - Event Logging - UR
MACI (CSMS)	<ul style="list-style-type: none"> - DCE 	<ul style="list-style-type: none"> - DCE Security Server 	<ul style="list-style-type: none"> - PF - ServerUR - Error Logging - Event Logging - UR

4.8.4 Distributed Computing Configuration Item Process Descriptions

Process descriptions are not applicable for the DCCI CSCI.

4.8.5 Distributed Computing Configuration Item Process Interface Descriptions

Process interface descriptions are not applicable for the DCCI CSCI.

4.8.6 Distributed Computing Configuration Item Data Stores

Data stores are not applicable for the DCCI CSCI.

4.8.6.1 Subscription Server Computer Software Component Description

4.8.6.1.1 Subscription Server Functional Overview

The Subscription Server (SBSRV) CSC provides the capability to register events, submit subscriptions, and process subscriptions upon event notification. Events and subscriptions are stored persistently in the SBSRV Database. During user registration, events are made available through the Interoperability Subsystem's (IOS) Advertisement Service. Subscriptions are submitted for an advertised event. Only users with a valid ECS user profile can submit subscriptions. The subscriptions can be qualified and can also include information specifying an action to be performed on behalf of the subscriber (e.g., acquire a data granule). Subscriptions can also be updated or deleted from the database. Upon event notification, all subscriptions for the event are extracted from persistent storage and associated actions are performed. Additionally, subscribers receive notification the event was triggered, via E-mail or through message passing (i.e., a message from a process). The SBSRV also includes an Operator GUI for entering, updating, and deleting subscriptions interactively.

4.8.6.1.2 Subscription Server Context

Figure 4.8.6.1.2-1 is the Subscription Server context diagram. Table 4.8.6.1.2-1 provides descriptions of the interface events in the Subscription Server context diagram.

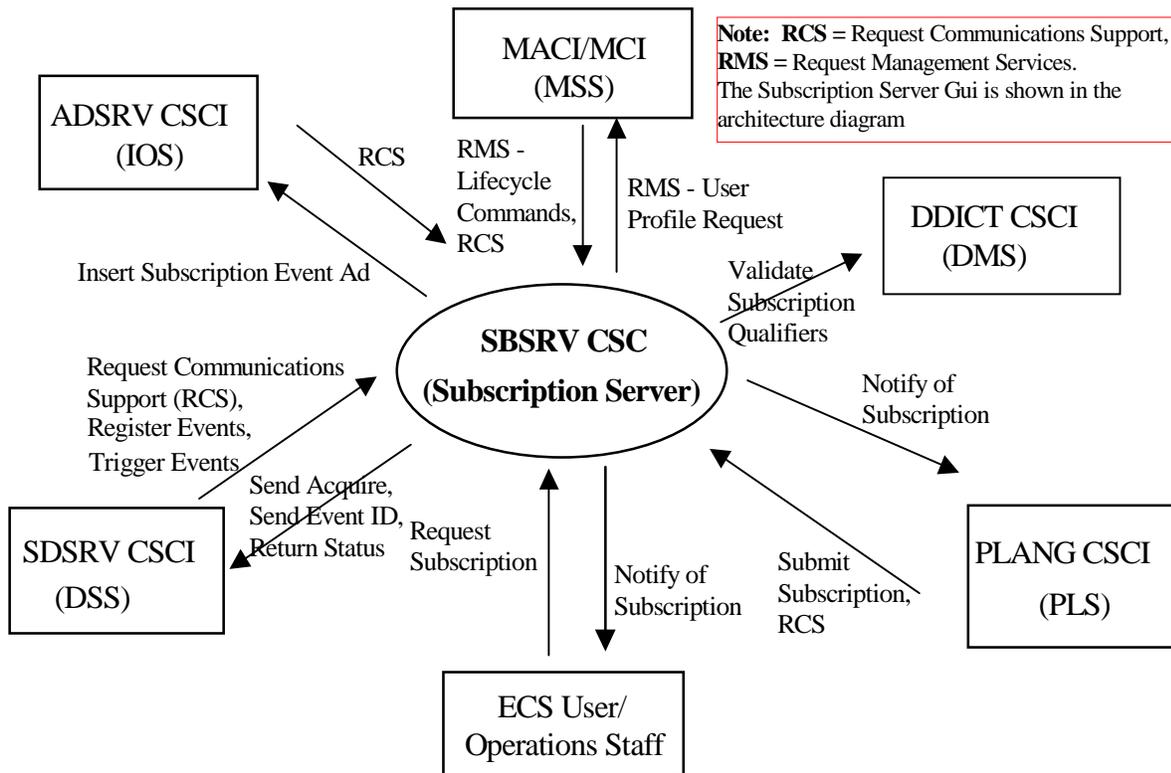


Figure 4.8.6.1.2-1. Subscription Server Context Diagram

Table 4.8.6.1.2-1. Subscription Server Interface Events

Event	Interface Event Description
Request Management Services	<p>The MCI and MACI provide a basic management library of services to the subsystems, implemented as client or server applications, using the DCCI CSCI Process Framework. The basic management library of services include:</p> <ul style="list-style-type: none"> • Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training). • User Profile Request - The MCI provides requesting CSCIs with User Profile parameters such as e-mail address and shipping address to support their processing activities.
Validate Subscription Qualifiers	The SBSRV CSC sends queries to the DDICT CSCI for type and range information to validate qualifiers.
Notify of Subscription	The SBSRV CSC sends notification (E-mail or inter-process) to the subscriber, Operations Staff, and the PLANG CSCI when the subscribed event occurs.
Submit Subscription	The PLANG CSCI submits a subscription request to the SBSRV CSC using the advertisement subscribing to an insert event for an ESDT.
Request Communications Support (RCS)	The DCCI CSCI provides a library of services available to each SDPS or CSMS CSCI. The services required performing the specific CSCI assignments are requested by the CSCI from the DCCI CSCI. These services include: Distributed Computing Environment (DCE) support, file transfer services, Network & Distributed File Services, Bulk Data transfer services, file copying services, name/address services, password services, Server Request Framework (SRF), Universal Reference (UR), Error/Event logging, message passing, Fault Handling services, User Authentication services, Mode Information, and retrieving the requested configuration attribute-value pairs from the Configuration Registry for ECS applications that request them.
Request Subscription	A subscriber (optionally) sends information with the subscription, specifying an action(s) (e.g., acquire or update) to be taken when the subscribed event occurs.
Send Acquire	An “acquire” (instruction to obtain data) is created by the DCCI CSCI and sent to the SDSRV CSCI via remote procedure call. This is similar to the “Request Product” interface event, except it applies to EDOS expedited data.
Send Event ID	The SBSRV CSC sends Event IDs to the SDSRV CSCI when ESDTs are installed or when ESDTs are updated by adding additional events.
Return status	Status returned by the DCCI CSCI to the SDSRV CSCI to simply indicate that the request was received, not that the action succeeded
Register Events	The SDSRV CSCI sends the subscription events for an Earth Science Data Type to the DCCI CSCI (Subscription Server) when an ESDT is installed into the system or when an ESDT is updated by adding additional events.
Trigger Events	The SDSRV CSCI notifies the DCCI CSCI (via an event trigger) when a subscription event occurs on an Earth Science Data Type Service.
Insert Subscription Event Ad	The ADSRV CSCI receives requests to insert subscription event advertisements from the SBSRV CSC.

4.8.6.1.3 Subscription Server Architecture

Figure 4.8.6.1.3-1 is the Subscription Server architecture diagram. The diagram shows the events sent to the Subscription Server process and the events the Subscription Server process sends to other processes.

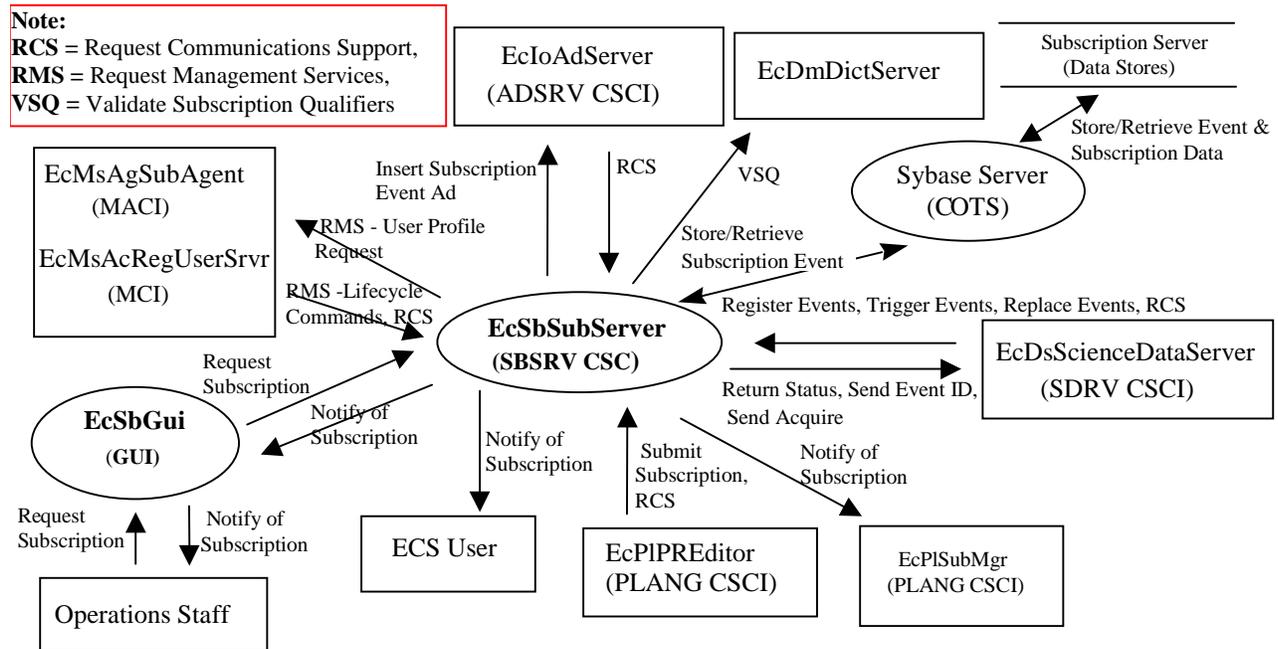


Figure 4.8.6.1.3-1. Subscription Server Architecture Diagram

4.8.6.1.4 Subscription Server Process Descriptions

Table 4.8.6.1.4-1 provides descriptions of the processes shown in the Subscription Server architecture diagram.

Table 4.8.6.1.4-1. Subscription Server Processes (1 of 2)

Process	Type	COTS/ Developed	Functionality
EcSbSubServer	Server	Developed	The Subscription Server enables an event producer to register and trigger events. A subscriber can submit subscriptions for an event. Events and subscriptions can also be updated and deleted.

Table 4.8.6.1.4-1. Subscription Server Processes (2 of 2)

Process	Type	COTS/ Developed	Functionality
EcSbGui	GUI	Developed	The Subscription GUI provides an operator interface for submitting, updating and deleting subscriptions.
Sybase Server	Server	COTS	The Sybase Server is the SQL Server for the Subscription Server and is only run by the DAAC Operations staff.

4.8.6.1.5 Subscription Server Process Interface Descriptions

Table 4.8.6.1.5-1 provides descriptions of the interface events shown in the Subscription Server architecture diagram.

Table 4.8.6.1.5-1. Subscription Server Process Interface Events (1 of 6)

Event	Event Frequency	Interface	Initiated by	Event Description
Request Communications Support	One service per request	<p><i>Library:</i> EcPf</p> <p><i>Classes:</i> EcPfManagedServer, EcPfclient</p> <p><i>Library(Common):</i> EcUr</p> <p><i>Class:</i> EcUrServerUR</p> <p><i>Library:</i> Event</p> <p><i>Class:</i> EcLgErrorMsg</p> <p><i>Process:</i> EcSbSubServer</p> <p><i>Library:</i> EcSbCI</p> <p><i>Classes:</i> EcCIEvent, EcCITriggerEventCb, EcCIRegisterEventCb</p>	<p>Process: EcDsScienceDataServer</p> <p><i>Classes:</i> DsDeEventCustomizer, DsBtSbSbrvNotifier</p> <p>Process: EcPIPREditor</p> <p><i>Library:</i> PICore1</p> <p><i>Classes:</i> Most PLS Classes</p> <p>Process: EcIoAdServer</p> <p><i>Libraries:</i> IoAdCore, IoAdSubs</p> <p><i>Classes:</i> IoAdApprovedAdv, IoAdGroup, IoAdProvider, IoAdProduct, IoAdSignatureServiceAdv</p> <p>Process: EcMsAgSubAgent,</p> <p><i>Library:</i> EcAgInstrm</p> <p><i>Class:</i> EcAgManager</p> <p>Process: EcMsAcRegUserSrvr</p> <p><i>Libraries:</i> MsAcCInt, MsAcComm</p> <p><i>Class:</i> MsAcUsrProfile</p>	<p>The DCCI CSCI Process Framework provides a library of services available to each SDPS or CSMS process. The services required to perform the specific process assignments are requested by the process from the Process Framework. These services include: Distributed Computing Environment (DCE) support, file transfer services, Network & Distributed File Services, Bulk Data transfer services, file copying services, name/address services, password services, Server Request Framework (SRF), Universal Reference (UR), Error/Event logging, message passing, Fault Handling services, User Authentication services, and Mode information.</p>

Table 4.8.6.1.5-1. Subscription Server Process Interface Events (2 of 6)

Event	Event Frequency	Interface	Initiated by	Event Description
Validate Subscription Qualifiers	Upon request	<i>Process:</i> EcDmDictServer <i>Library:</i> EcDmDdClient <i>Class:</i> DmDdCIRequest	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbSr <i>Class:</i> EcSbSubscription	The EcSbSubServer sends queries to the EcDmDictServer for type and range information to validate qualifiers.
Store/Retrieve Subscription Event	One per store and retrieve event	Sybase Server (COTS)	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbSr <i>Classes:</i> EcSbEventStore, EcSbSubscriptionStore	The EcSbSubServer stores and retrieves subscription information and events from the Subscription Server Data Stores via the Sybase Server.
Store/Retrieve Event & Subscription Data	One per request	Sybase Server (COTS)	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbSr <i>Classes:</i> EcSbEventStore, EcSbSubscriptionStore	The EcSbSubServer stores and retrieves event and subscription data via the Sybase Server in persistent data storage tables in the Subscription Server (Data Stores). For an explanation of this data, see the 'Subscription Server Data Stores' subsection.
Register Events	One per event	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbCl <i>Classes:</i> EcCIEvent, EcCIRegisterEventCb	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsDelsh <i>Class:</i> DsDeEventCustomizer	The EcDsScienceDataServer sends the subscription events for an Earth Science Data Type to the EcSbSubServer when an ESDT is installed into the system or when an ESDT is updated by adding additional events.
Trigger Events	One per event trigger	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbCl <i>Classes:</i> EcCIEvent, EcCITriggerEventCb	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsBtSh <i>Class:</i> DsBtSbsrvNotifier	The EcDsScienceDataServer notifies the EcSbSubServer (via an event trigger) when a subscription event occurs on an Earth Science Data Type Service.

Table 4.8.6.1.5-1. Subscription Server Process Interface Events (3 of 6)

Event	Event Frequency	Interface	Initiated by	Event Description
Replace Events	One per ESDT update	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbSrSh <i>Class:</i> EcCIEvent	<i>Process:</i> EcDsScienceData Server <i>Class:</i> DsDeEventCustomizer	The EcDsScienceDataServer sends the updated subscription events for an Earth Science Data Type (ESDT) to the EcSbSubServer when an ESDT is updated in the system. This event replaces the original event in the EcSbSubServer.
Return Status	One per request	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsBt <i>Class:</i> DsBtSbsrvNotifier	<i>Process:</i> EcSbSubServer <i>Library:</i> EcUt	Status returned by the EcSbSubServer to the EcDsScienceDataServer to simply indicate that the request was received, not that the action succeeded.
Send Event ID	One per ESDT install	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsDelSh <i>Class:</i> DsDeDataDictController	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbCI <i>Class:</i> EcCIEvent	The EcSbSubServer sends Event IDs to the EcDsScienceDataServer when ESDTs are installed or when ESDTs are updated by adding additional events.
Send Acquire	One per request	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsCI <i>Classes:</i> DsCIRequest, DsCICommand, DsCIESDTReferenceCollector	<i>Process:</i> EcCsEmailParser <i>Class:</i> EcCsEmailParser <i>Process:</i> EcSbSubServer <i>Library:</i> EcSbSr <i>Class:</i> EcSbSubscription	An “acquire” (instruction to obtain data) is created by the EcCsEmailParser or EcSbSubServer and sent to the EcDsScienceDataServer via remote procedure call. This is similar to the “Request Product” interface event, except it applies to EDOS expedited data.
Notify of Subscription	One per subscription submitted	<i>Process:</i> EcPISubMgr <i>Class:</i> PISubMsgCb <i>Process:</i> EcSbGui <i>Class:</i> EcSbSubscriptionDispatcher	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbSr <i>Classes:</i> EcSbSubscription, EcSbNotification	The EcSbSubServer sends E-mail to the ECS User, Operations Staff (via EcSbGui), or inter-process notification (via the message-passing framework) to the EcPISubMgr.

Table 4.8.6.1.5-1. Subscription Server Process Interface Events (4 of 6)

Event	Event Frequency	Interface	Initiated by	Event Description
Submit Subscription	One per subscription with acquire request	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbCI <i>Class:</i> EcCISubscription	<i>Process:</i> EcPIPREditor_IF <i>Library:</i> PICore1 <i>Class:</i> PIDataType	The EcPIPREditor_IF sends to the EcSbSubServer information with the subscription submitted by an ECS User or the Operations staff, specifying an action(s) (e.g., acquire or update) to be taken when the subscribed event occurs.
Request subscription	One per subscription submitted	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbSr <i>Classes:</i> EcSbSubmitSubRequest, EcSbSubscription	Operations Staff <i>Process:</i> EcSbGui <i>Library:</i> EcSbCI <i>Class:</i> EcCISubscription	The Operations Staff can make a request for a subscription to the EcSbSubServer via the EcSbGui on behalf of an ECS User.

Table 4.8.6.1.5-1. Subscription Server Process Interface Events (6 of 6)

Event	Event Frequency	Interface	Initiated by	Event Description
Insert Subscription Event Ad	One per event registration	<i>Process:</i> EcIoAdServer <i>Libraries:</i> IoAdCore, IoAdSubs <i>Classes:</i> IoAdSignatureServiceAdv, IoAdApprovedAdv, IoAdGroup, IoAdProvider	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbSr <i>Class:</i> EcSbEvent	Upon event registration, the EcSbSubServer sends the event information along to the EcIoAdServer to post the event as valid for subscriptions.

4.8.6.1.6 Subscription Server Data Stores

Subscription Server uses the COTS software Sybase database for its persistent storage. The following is a brief description of the types of data contained in the database:

- **event data:** includes event type, user id, qualified metadata attribute names, and other information describing an event
- **subscription data:** includes a link to the event data, user id, start and expiration dates, qualified metadata values (optional), and action information (optional) for what to do when an event occurs
- **persistence data:** include three tables, which store temporary data for uniquely identify a trigger call, the triggered subscriptions and actions associated with it. These tables are used to avoid lost or duplicated triggers. The Subscription Server processes these tables during WARM restart and discards them in COLD start.

Table 4.8.6.1.6-1 provides descriptions of the data found in the seven separate Sybase data stores used by the Subscription Server. More detail on these data stores can be found in the Subscription Server Database Design and Schema Specifications for the ECS Project (Refer to CDRL 311).

Table 4.8.6.1.6-1. Subscription Server Data Stores (1 of 2)

Data Store	Type	Functionality
EcSbEvent	Sybase	Contains the list of events to which a user or another subsystem can subscribe.
EcSbNewEventID	Sybase	This data store contains the next available ID for the EcSbEvent table.
EcSbNewSubID	Sybase	This data store contains the next available ID for the EcSbSubscription table.

Table 4.8.6.1.6-1. Subscription Server Data Stores (2 of 2)

Data Store	Type	Functionality
EcSbSubscription	Sybase	This data store lists all the user and subsystem subscriptions. Each event can have many subscriptions. Each user can have many subscriptions. The same user can subscribe to the same event with different constraints. It is also possible that a user could subscribe to the same event with the same constraints.
EcSbTriggerRequest	Sybase	This data store contains the entire trigger request from Science Data Server in a predefined period of time, including RpcID, EventID, Actual (the actual qualifier list of the trigger request), TimeReceived (time the request was received) and EventStatus (status of the trigger request). This table will stay for a configurable amount of time.
EcSbSubWorkOff	Sybase	This data store contains all the temporary data of subscriptions on the triggered events. It includes RpcID, SubID and TimeReceived. It provides a link between the subscriptions and the given event trigger request.
EcSbActionWorkOff	Sybase	This data store contains all the actions of triggered subscriptions that are still in processing. It includes ActionID (uniquely identify the action, RpcID, SubID, TimeReceived, Tries, OutBoundRpcID and ActionStatus.

4.8.6.2 ASTER DAR Gateway Server Software Description

4.8.6.2.1 ASTER DAR Gateway Server Functional Overview

The ASTER DAR Gateway Server provides interoperability between the CSS MOJO Gateway and the DAR API with an interface to the ASTER GDS servers.

The DAR API provides the functionality to transmit data concerning the DAR between the DAR Gateway and the DAR Server and makes the DAR Server database information available to ECS users. The functionality is provided to support five DAR APIs: SubmitDAR, ModifyDAR, queryxARContents, queryxARSummary, and queryxARScenes. DAR Communications are part of the ECS and ASTER GDS interface, where ground support for mission operations and science data processing are provided for the ASTER instrument on-board the EOS AM-1 spacecraft. The DAR Server is located in Japan and transparently interacts with ASTER Operations Segment (AOS) xAR Server and xAR Database to provide data to its clients. The DAR Server provides ECS users access to DAR database information via an API. DAR-related communication between ECS and the ASTER GDS is through ASTER GDS provided APIs, integrated into the DAR Communications Gateway. (The DAR Communications Gateway server is located at the EROS Data Center (EDC).)

4.8.6.2.2 ASTER DAR Gateway Server Context

Figure 4.8.6.2.2-1 is the ASTER DAR Gateway Server context diagram and Table 4.8.6.2.2-1 provides descriptions of the interface events shown in the ASTER DAR Gateway context diagram. The information contained in the context diagram and interface events table is,

respectively, applicable to each of the ASTER DAR Gateway functions: SubmitDAR, ModifyDAR, queryxARContents, queryxARSummary, and queryxARScenes.

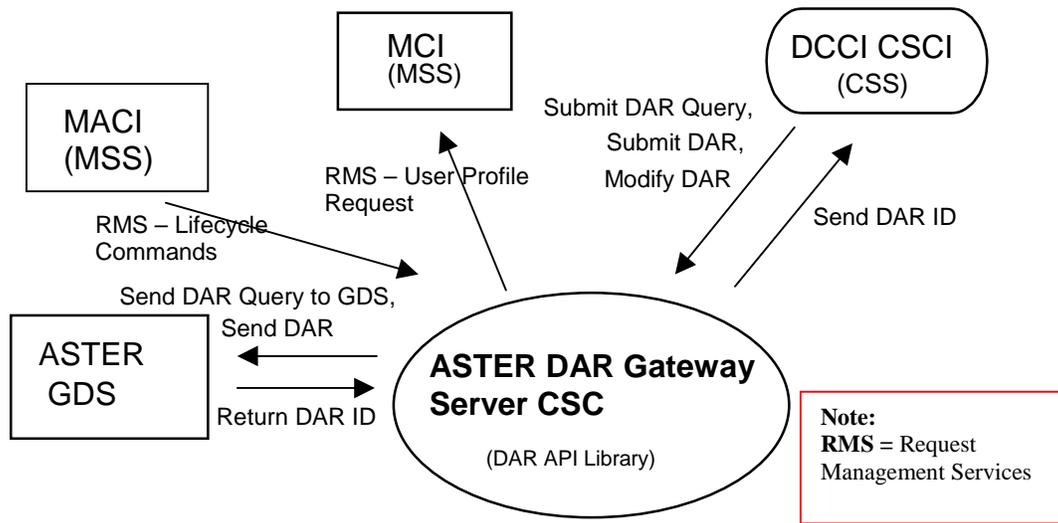


Figure 4.8.6.2.2-1. ASTER DAR Gateway Server Context Diagram

Table 4.8.6.2.2-1. ASTER DAR Gateway Server Interface Events (1 of 2)

Event	Interface Event Description
Submit DAR Query	The ECS MOJO Gateway sends a DAR query, received from the ASTER JAVA DAR Tool and of type queryxARContents, queryxARSummary, or queryxARScenes, to the ECS ASTER DAR Gateway Server via DCE/RPCs.
Submit DAR	The ECS MOJO Gateway submits DARs, received from the ASTER JAVA DAR Tool, to the ECS ASTER DAR Gateway Server via DCE/RPCs.
Modify DAR	The ECS MOJO Gateway sends modified DARs, received from the ASTER JAVA DAR Tool, to the ECS ASTER DAR Gateway Server via DCE/RPCs.
Send DAR ID	The ASTER DAR Gateway Server extracts the returned DAR ID and sends it to the Java DAR Tool, via the MOJO Gateway Server.
Send DAR Query to GDS	The ECS user sends the DAR query to the ASTER GDS DAR Server by DAR API library calls, which communicate via TCP/IP sockets over the EBnet.
Send DAR	The ECS user sends the request to the ASTER GDS Storage Server by DAR API library calls which communicates via TCP/IP sockets over EBnet. The ASTER GDS returns a DAR ID to the ASTER DAR Gateway Server at the ECS. The ASTER DAR Gateway Server extracts the returned DAR ID and sends it to the ASTER DAR tool.
Return DAR ID	The ASTER GDS returns a DAR ID to the ASTER DAR Gateway Server at the ECS.

Table 4.8.6.2.2-1. ASTER DAR Gateway Server Interface Events (2 of 2)

Event	Interface Event Description
Request Management Services	<p>The MCI and MACI provide a basic management library of services to the CSCIs, implemented as client or server applications, using the DCCI CSCI Process Framework. The basic management library of services include:</p> <ul style="list-style-type: none"> Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training). User Profile Request - The MCI provides requesting CSCIs with User Profile parameters such as e-mail address and shipping address to support their processing activities.

4.8.6.2.3 ASTER DAR Gateway Server Architecture

Figure 4.8.6.2.3-1 is the ASTER DAR Gateway Server architecture diagram. The diagram shows the events sent to the ASTER DAR Gateway Server process and the events the ASTER DAR Gateway Server process sends to other processes.

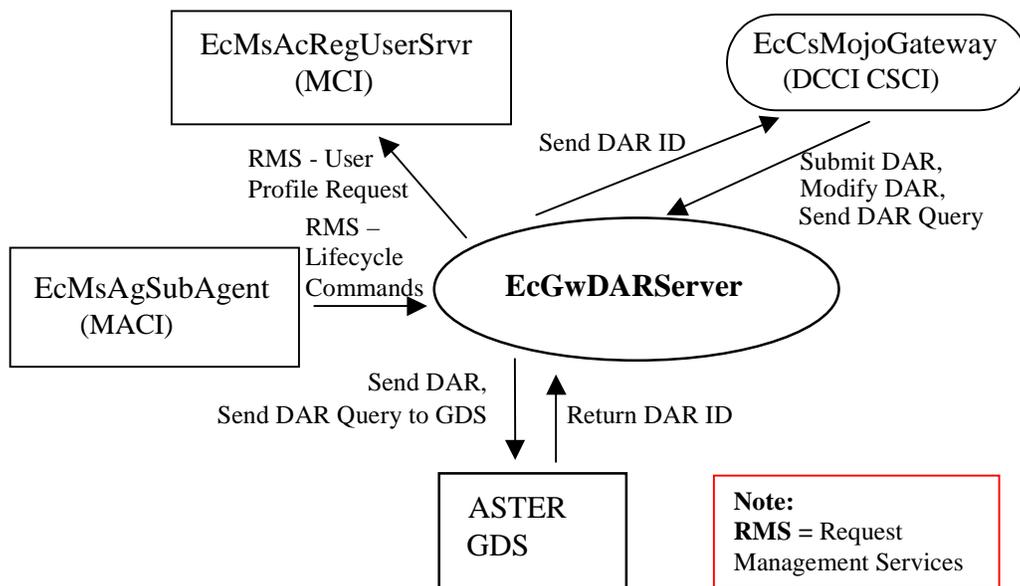


Figure 4.8.6.2.3-1. ASTER DAR Gateway Server Architecture Diagram

4.8.6.2.4 ASTER DAR Gateway Server Process Descriptions

Table 4.8.6.2.4-1 provides descriptions of the processes in the ASTER DAR Gateway Server architecture diagram.

Table 4.8.6.2.4-1. ASTER DAR Gateway Server Processes

Process	Type	COTS/ Developed	Functionality
EcGwDARServer	Server	COTS	<p>The ASTER DAR Gateway Server provides five functions.</p> <ul style="list-style-type: none"> • Submit DAR function: Registered users use the Java DAR tool to create a DAR. The DAR client sends the DAR to the Mojo Gateway server and gets back a DAR ID. • Modify DAR function: A Modified DAR is sent from the DAR client to the Mojo Gateway server and gets a status back. Registered users use the DAR tool to “Modify” an existing request and “Submit” the modified request in the default synchronous mode. • QueryxARContents: Gets xAR contents by matching xarID. A registered user changes the default mode to synchronous and submits the modified DAR. • QueryxARSummary: Gets a subxAR status from the AOS xAR DB by matching xAR IDs. This function responds to multiple subxAR statuses for one request. • QueryxARScenes: Gets multiple xAR summaries from the AOS xAR DB by matching the search condition. This function responds to multiple xAR summaries for one request. <p>Synchronous request processing is supported. Asynchronous request processing is supported. Multiple concurrent requests are supported.</p>

4.8.6.2.5 ASTER DAR Gateway Server Process Interface Descriptions

Table 4.8.6.2.5-1 provides the descriptions of the interface events shown in the ASTER DAR Gateway Server architecture diagram.

Table 4.8.6.2.5-1. ASTER DAR Gateway Server Process Interface Events (1 of 3)

Event	Event Frequency	Interface	Initiated by	Event Description
Send DAR ID	One per DAR ID sent	<i>Process:</i> EcCsMojoGateway <i>Library:</i> EcGwDAR <i>Class:</i> EcGwSubmitDARRequest_C	<i>Process:</i> EcGwDARServer <i>Class:</i> EcGwDARGatewayRequest_S	The EcGwDARServer extracts the returned DAR ID and sends it to the Java DAR Tool, via the EcCsMojoGateway.
Submit DAR	One per DAR	<i>Process:</i> EcGwDARServer <i>Library:</i> EcGwDAR <i>Class:</i> EcGwDARSubmitDarRequest_C	User <i>Process:</i> EcCsMojoGateway <i>Class:</i> EcMjDARSubmitDarProxy	A user submits a DAR, through the EcCsMojoGateway, to the ECS EcGwDARServer via DCE RPCs.
Modify DAR	One per modified DAR	<i>Process:</i> EcGwDARServer <i>Library:</i> EcGwDAR <i>Class:</i> EcGwDARModifyDarRequest_C	<i>Process:</i> EcCsMojoGateway <i>Class:</i> EcMjDARModifyProxy	A user modifies a DAR and sends the modified request, through the EcCsMojoGateway, to the ECS EcGwDARServer via DCE RPCs.
Send DAR Query	One per query request	<i>Process:</i> EcGwDARServer <i>Library:</i> EcGwDAR <i>Classes:</i> EcGwDARQueryxARContentsRequest_C, EcGwDARQueryxARSummaryRequest_C EcGwDARQueryxARScenesRequest_C	<i>Process:</i> EcCsMojoGateway <i>Class:</i> EcMjDARQueryxARContentsProxy, EcMjDARQueryxARSummaryProxy, EcMjDARQueryxARScenesProxy	The EcCsMojoGateway sends the query request, received from the ASTER JAVA DAR Tool, to the ECS ASTER DAR Gateway Server via DCE RPCs.
Return DAR ID	One per DAR ID return	<i>Process:</i> EcGwDARServer <i>Library:</i> IcDarApi	ASTER GDS	The ASTER GDS returns a DAR ID to the EcGwDARServer at the ECS.

Table 4.8.6.2.5-1. ASTER DAR Gateway Server Process Interface Events (2 of 3)

Event	Event Frequency	Interface	Initiated by	Event Description
Send DAR	One per ASTER DAR send	ASTER GDS	<i>Process:</i> EcGwDARServer <i>Library:</i> IcDarApi <i>Class:</i> EcGwDARGatewayRequest_S	The EcGwDARServer sends the request to the ASTER GDS Storage Server via DAR API library, which communicates via TCP/IP sockets over the EBnet. The ASTER GDS returns a DAR ID to the EcGwDARServer at the ECS. The EcGwDARServer extracts the returned DAR ID and sends it to the ECS DAR tool, via the EcCsMojoGateway.
Send DAR Query to GDS	One per query request	ASTER GDS	<i>Process:</i> EcGwDARServer <i>Library:</i> IcDarApi <i>Class:</i> EcGwDARGatewayRequest_S	The ASTER DAR Gateway Server sends the query to the ASTER GDS DAR Server via DAR API Library calls, which communicates via TCP/IP sockets over the EBnet.

Table 4.8.6.2.5-1. ASTER DAR Gateway Server Process Interface Events (3 of 3)

Event	Event Frequency	Interface	Initiated by	Event Description
Request Management Services	One per service request	<p><i>Process:</i> EcGwDARServer</p> <p><i>Process:</i> EcMsAcRegUserSrvr</p> <p><i>Libraries:</i> MsAcCInt, MsAcComm</p> <p><i>Class:</i> EcAcProfileMgr</p>	<p><i>Process:</i> EcMsAgSubAgent</p> <p><i>Library:</i> EcAgInstrm</p> <p><i>Class:</i> EcAgManager</p> <p><i>Process:</i> EcGwDARServer</p> <p><i>Class:</i> EcGwDARGatewayRequest_S</p>	<p>The EcMsAgSubAgent and EcMsAcRegUserSrvr provide a basic management library of services to the subsystems, implemented as client or server applications, using the DCCI CSCI Process Framework. The basic management library of services includes:</p> <ul style="list-style-type: none"> • Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training). • User Profile Request - The EcMsAcRegUserSrvr provides requesting processes with User Profile parameters such as e-mail address and shipping address to support their processing activities.

4.8.6.2.6 ASTER DAR Gateway Server Data Stores

Data stores are not applicable for the ASTER DAR Gateway.

4.8.6.3 E-mail Parser Gateway Server Software Description

4.8.6.3.1 E-mail Parser Gateway Server Functional Overview

Expedited Data Sets (EDS) are raw satellite telemetry data processed into time-ordered instrument packets with packets separated into files for a given downlink contact. The ECS provides EDS to the ASTER GDS to use in evaluating the operation of the instrument. Level 0 EDS produced at the DAAC are staged for up to 48 hours before delivery to investigators at the Science Computing Facilities.

The E-mail Parser Gateway Server forwards notifications to the ASTER GDS when a EDS is received (the notification is called an EDN) and processes E-mail messages from the ASTER GDS requesting delivery of an EDS (the messages are EDRs). To facilitate this, EDS subscriptions are placed at the GSFC DAAC by user services personnel on behalf of the ASTER GDS. Each time the GSFC DAAC receives a EDS from EDOS, the subscription is triggered and an E-mail message is sent to the ASTER GDS. The subscription notifications are sent to the E-mail Parser Gateway to turn them into properly formatted EDN mail messages and sends them to the ASTER GDS via the MSS ASTER E-mail header handler to have the appropriate mail header information added. When ASTER orders the EDS, an E-mail message is sent via the MSS ASTER E-mail header handler to the E-mail Parser Gateway. The E-mail Parser Gateway formulates and submits the corresponding acquire request to the DSS SDSRV CSCI for an FTP push distribution of the EDS to ASTER.

4.8.6.3.2 E-mail Parser Gateway Server Context

Figure 4.8.6.3.2-1 is the E-mail Parser Gateway Server context diagram. Table 4.8.6.3.2-1 provides descriptions of the interface events shown in the E-mail Parser Gateway Server context diagram.

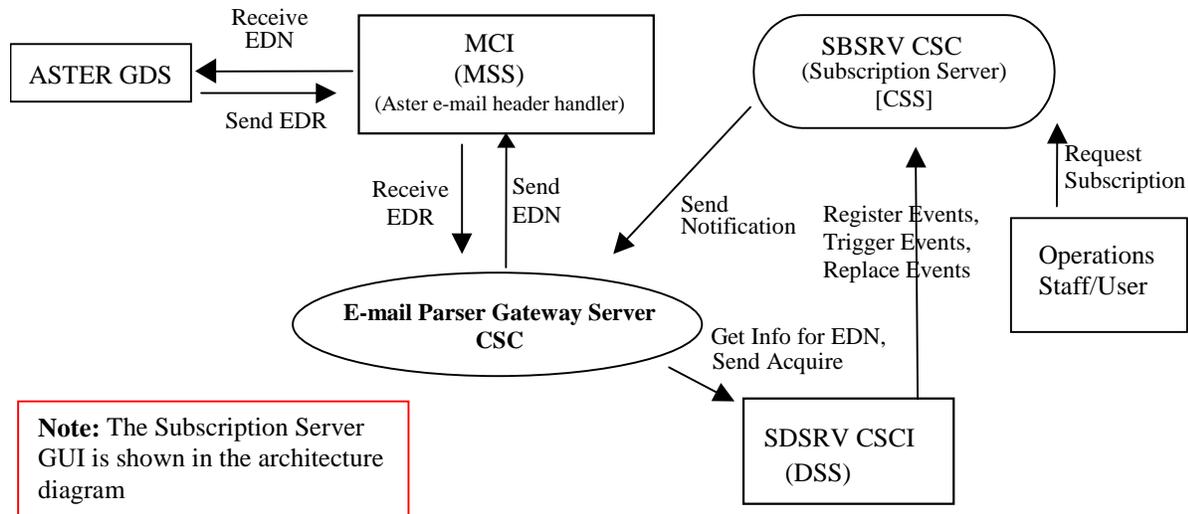


Figure 4.8.6.3.2-1. E-mail Parser Gateway Server Context Diagram

Table 4.8.6.3.2-1. E-mail Parser Gateway Server Interface Events

Event	Interface Event Description
Send EDN	The E-mail Parser Gateway Server CSC stores the EDN messages with URs, time range, etc., and sends the EDN to the MCI.
Send Notification	The SBSRV CSC sends a notification via E-mail to the E-mail Parser Gateway Server CSC to notify the ASTER GDS of the arrival of EDS from EDOS to the ECS.
Register Events	The SDSRV CSCI sends the subscription events for an Earth Science Data Type to the DCCI CSCI (Subscription Server) when an ESDT is installed into the system or when an ESDT is updated by adding additional events.
Trigger Events	The SDSRV CSCI notifies the DCCI CSCI (via an event trigger) when a subscription event occurs on an Earth Science Data Type Service.
Replace Events	The SDSRV CSCI sends the updated subscription events with modified qualifiers for an Earth Science Data Type (ESDT) to the DCCI CSCI (Subscription Server) when an ESDT is updated. This event replaces the original event in the DCCI CSCI.
Request Subscription	DAAC Operations staff place a subscription with the subscription server (SBSRV CSC), on behalf of the ASTER GDS, once in the beginning of the mission and/or once at a time defined in an Operations Agreement between the ASTER GDS and the ECS.
Get Info for EDN	Expedited Data Set Notification (EDN) information is obtained from the SDSRV CSCI, by request, and used by the DCCI CSCI to send messages to users at the ASTER GDS.
Send Acquire	An “acquire” (instruction to obtain data) is created by the DCCI CSCI and sent to the SDSRV CSCI via remote procedure call. This is similar to the “Request Product” interface event, except it applies to EDOS expedited data.
Receive EDN	The MCI E-mail header handler adds a header to the EDN and is received by the ASTER GDS via E-mail over EBnet.
Send EDR	ASTER GDS personnel select the EDN needed and send an EDR to the MCI.
Receive EDR	The MCI E-mail header handler strips the EDR header and is received by the E-mail Parser Gateway Server CSC.

4.8.6.3.3 E-mail Parser Gateway Server Architecture

Figure 4.8.6.3.3-1 is the E-mail Parser Gateway Server architecture diagram. The diagram shows the events sent to the E-mail Parser Gateway Server process and the events the E-mail Parser Gateway Server process sends to other processes.

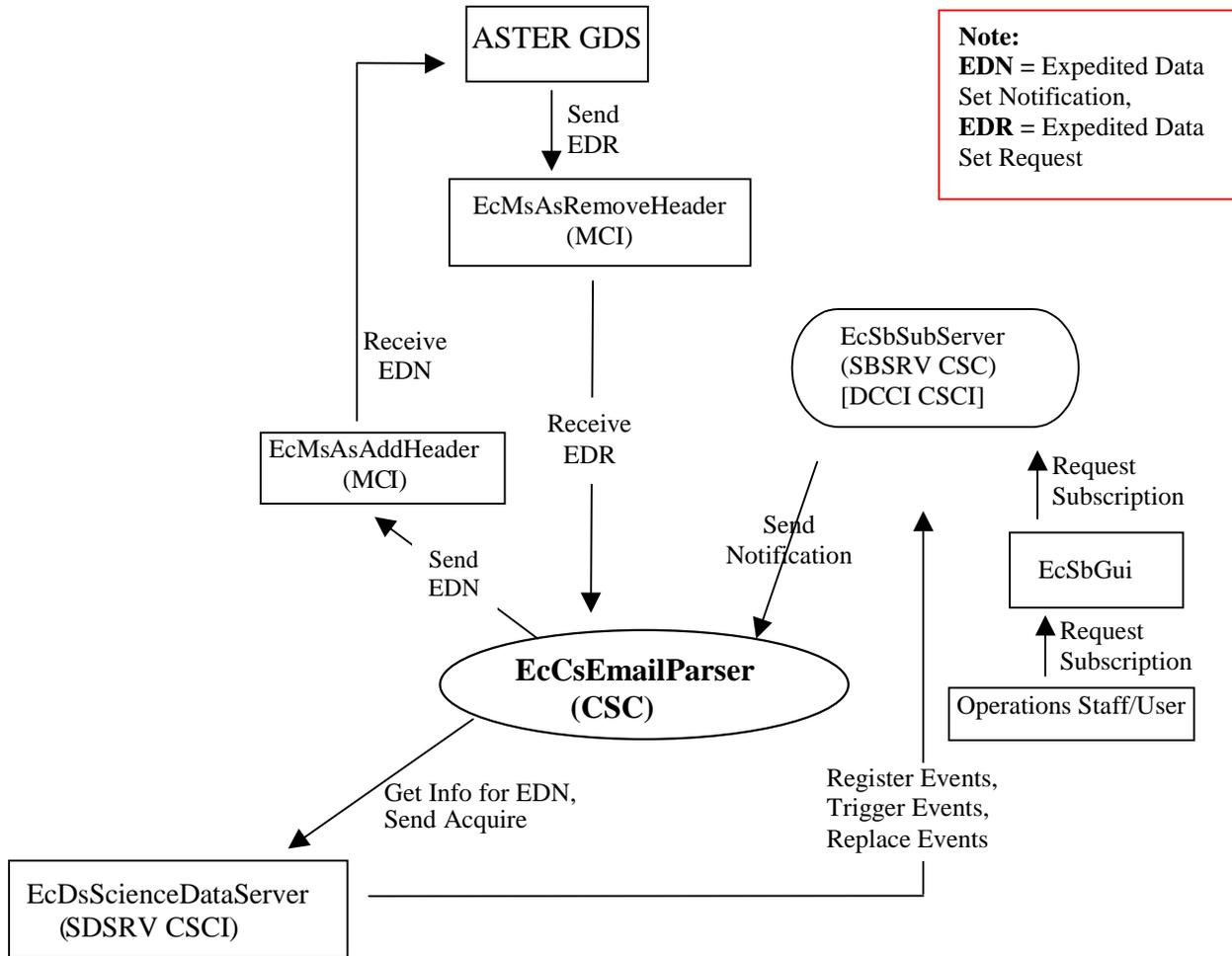


Figure 4.8.6.3.3-1. E-mail Parser Gateway Server Architecture Diagram

4.8.6.3.4 E-mail Parser Gateway Server Process Descriptions

Table 4.8.6.3.4-1 provides a description of the process shown in the E-mail Parser Gateway Server architecture diagram.

Table 4.8.6.3.4-1. E-mail Parser Gateway Server Processes

Process	Type	Cots/Developed	Functionality
EcCsEmailParser	Server	Developed	<ul style="list-style-type: none"> • Get UR from the Subscription Notification and use this UR to get the information for EDN from the EcDsScienceDataServer and send it to the EcMsAsAddHeader for notifying ASTER GDS. • Get EDR from the EcMsAsRemoveHeader. • Store and parse subscriptions and EDR in /EDN, /EDR directory • Send an Acquire request to the EcDsScienceDataServer via an RPC for an EDR request.

4.8.6.3.5 E-mail Parser Gateway Server Process Interface Descriptions

Table 4.8.6.3.5-1 provides descriptions of the interface events shown in the E-mail Parser Gateway Server architecture diagram.

Table 4.8.6.3.5-1. E-mail Parser Gateway Server Process Interface Events (1 of 3)

Event	Event Frequency	Interface	Initiated by	Event Description
Send EDR	One per EDR send	<i>Script:</i> EcMsAsRemoveHeader	ASTER GDS Operations Staff <i>Process:</i> EcCsEmailParser	After selecting the EDN, the ASTER GDS personnel send an EDR to the EcMsAsRemoveHeader to have the header removed.
Receive EDR	One per e-mail send from ASTER GDS	<i>Script:</i> EcMsAsRemoveHeader	ASTER GDS (Operations Staff)	The ASTER GDS sends an EDR to the EcMsAsRemoveHeader script to remove the e-mail header and forward the e-mail to the EcCsEmailParser.
Send Notification	One per send of EDN	<i>Process:</i> EcCsEmailParser <i>Class:</i> EcCsEmailParser	<i>Process:</i> EcSbSubServer <i>Library:</i> CsEmMailRelA <i>Class:</i> CsEmMailRelA	The EcSbSubServer sends an EDN via E-mail to the EcCsEmailParser.

Table 4.8.6.3.5-1. E-mail Parser Gateway Server Process Interface Events (2 of 3)

Event	Event Frequency	Interface	Initiated by	Event Description
Request Subscription	One per subscription request	<i>Process:</i> EcSbSubServer <i>Libraries:</i> EcSbSr, EcSbCl <i>Classes:</i> EcSbSubmitSubRequest, EcSbSubscription, EcClSubscription	Operations Staff/User <i>Process:</i> EcSbGui <i>Class:</i> EcSbSubscriptionDispatcher	The Operations Staff subscribe to the ECS, via the EcSbSubServer, on behalf of the ASTER GDS. An ECS User can make a request for a subscription to the EcSbSubServer.
Register Events	One per ESDT installation	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbSrSh <i>Class:</i> EcSbEvent	<i>Process:</i> EcDsScienceDataServer <i>Class:</i> DsDeEventCustomizer	The EcDsScienceDataServer sends the subscription events for an Earth Science Data Type to the EcSbSubServer when an ESDT is installed into the system or when an ESDT is updated by adding additional events.
Trigger Events	One per trigger event	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbCl <i>Class:</i> EcClEvent	<i>Process:</i> EcDsScienceDataServer <i>Class:</i> DsBtSbsrvNotifier	The EcDsScienceDataServer notifies the EcSbSubServer (via an event trigger) when a subscription event occurs on an Earth Science Data Type Service.
Replace Events	One per ESDT update	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbSrSh <i>Class:</i> EcClEvent	<i>Process:</i> EcDsScienceDataServer <i>Class:</i> DsDeEventCustomizer	The EcDsScienceDataServer sends the updated subscription events for an Earth Science Data Type (ESDT) to the EcSbSubServer when an ESDT is updated. This event replaces the original event in the EcSbSubServer.
Get Info for EDN	One per notification of the ASTER GDS	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsCl <i>Class:</i> DsClESDTReference	<i>Process:</i> EcCsEmailParser <i>Class:</i> EcCsEmailParser	Expedited Data Set Notification (EDN) information is obtained from the EcDsScienceDataServer, by request, and used by the EcCsEmailParser to send messages to users at the ASTER GDS.

Table 4.8.6.3.5-1. E-mail Parser Gateway Server Process Interface Events (3 of 3)

Event	Event Frequency	Interface	Initiated by	Event Description
Send Acquire	One per acquire created	<i>Process:</i> EcDsScienceDataServer <i>Library:</i> DsCI <i>Class:</i> DsCIRequest	<i>Process:</i> EcCsEmailParser <i>Class:</i> EcCsEmailParser	An “acquire” (instruction to obtain data) is created by the EcCsEmailParser and sent to the EcDsScienceDataServer via remote procedure call. This is similar to the “Request Product” interface event, except it applies to EDOS expedited data.
Receive EDN	One per e-mail send from ECS	ASTER GDS	<i>Script:</i> EcMsAsAddHeader	The EcMsAsAddHeader script adds a header to the e-mail and forwards the e-mail to the ASTER GDS.
Send EDN	One per E-mail send	<i>Script:</i> EcMsAsAddHeader	<i>Process:</i> EcCsEmailParser <i>Class:</i> EcCsEmailParser	The EcCsEmailParser sends the Send EDN to the EcMsAsAddHeader to have a header added.

4.8.6.3.6 E-mail Parser Gateway Server Data Stores

Data Stores are not applicable for the E-mail Parser Gateway.

4.8.6.4 Landsat 7 Gateway Server Software Description

4.8.6.4.1 Landsat 7 Gateway Server Functional Overview

The ECS user interface provides access to the Landsat 7 Processing System (LPS) data collected with the Enhanced Thematic Mapper Plus (ETM+) instrument on the Landsat 7 satellite. The Landsat 7 Project reformats the raw instrument data into Level 0R data and provides the data to ECS for ingest, archive and distribution. All ECS registered users are permitted access to Level 0R data, metadata, and browse data archived by the ECS.

Because LPS is not DCE compatible, the Landsat 7 Gateway Server is used as an interface between the LPS and DCE/Object Oriented DCE (OODCE) based ECS services. This gateway provides the capabilities for the following activities:

- Automated data transfer from LPS to ECS requiring transmission of control messages to provide the file information and handshaking required to complete the data transfer
- LPS sends Level 0R data, associated inventory metadata, and browse data to ECS
- ECS sends an acknowledgment to LPS, after archiving the Landsat 7 data
- LPS sends data to ECS for ingesting, storing, and distributing for Pre-launch checkout of instruments and development of initial calibration information
- ECS interface testing, operations testing, and acceptance testing activities with LPS

- ECS ingesting, archiving, and acknowledging receipt of Level 0R data from LPS for the previous 12-hour period, within 8 hours of the receipt of the data availability notice (DAN) from the LPS.

4.8.6.4.2 Landsat 7 Gateway Server Context

Figure 4.8.6.4.2-1 is the Landsat 7 Gateway Server context diagram. Table 4.8.6.4.2-1 provides descriptions of the interface events shown in the Landsat 7 Gateway context diagram.

Using sockets for exchanging messages, the Landsat 7 Gateway Server receives the Data Availability Notice (DAN) from LPS when it has data for ingest. After the data is ingested and delivered, LPS sends a Data Delivery Acknowledgment (DDA) back to the Landsat 7 Gateway Server. Also using sockets, the gateway forwards the Data Availability Acknowledgment (DAA) and the Data Delivery Notice (DDN) from the INS to the LPS.

Using an RPC to exchange data status, the Landsat 7 Gateway Server receives DAA and DDN from the INS. The Landsat 7 Gateway forwards the DAN and the DDA to the INS.



Figure 4.8.6.4.2-1. Landsat 7 Gateway Server Context Diagram

Table 4.8.6.4.2-1. Landsat 7 Gateway Server Interface Events (1 of 2)

Event	Interface Event Description
DAA	The Data Availability Acknowledgment is forwarded by the ECS Landsat 7 Gateway Server to the Landsat 7 Processing System using sockets.
DDN	The Data Delivery Notice is forwarded by the ECS Landsat 7 Gateway Server to the Landsat 7 Processing System using sockets.
Send DAN	The DAN originated by the LPS is sent to the ECS Landsat 7 Gateway Server via sockets.
Send DDA	The DDA originated by the LPS is sent to the ECS Landsat 7 Gateway Server via sockets and is forwarded to the INS by the Landsat 7 Gateway using a RPC.
DAN	The Data Availability Notice is forwarded to the INS by the ECS Landsat 7 Gateway Server using a RPC.
DDA	The Data Delivery Acknowledgment is forwarded to the INS by the ECS Landsat 7 Gateway Server using a RPC.

Table 4.8.6.4.2-1. Landsat 7 Gateway Server Interface Events (2 of 2)

Event	Interface Event Description
Send DAA	The DAA originated by the INS is sent to the Landsat 7 Gateway via RPC.
Send DDN	The DDN originated by the INS is sent to the Landsat 7 Gateway via RPC.

4.8.6.4.3 Landsat 7 Gateway Server Architecture

The Landsat 7 Gateway Server is one ECS developed process, EcCsLandsat7Gateway. The Landsat 7 Gateway Server is managed via the Process Framework mechanism and Figure 4.8.6.4.3-1 is the architecture diagram of the Landsat 7 Gateway Server.

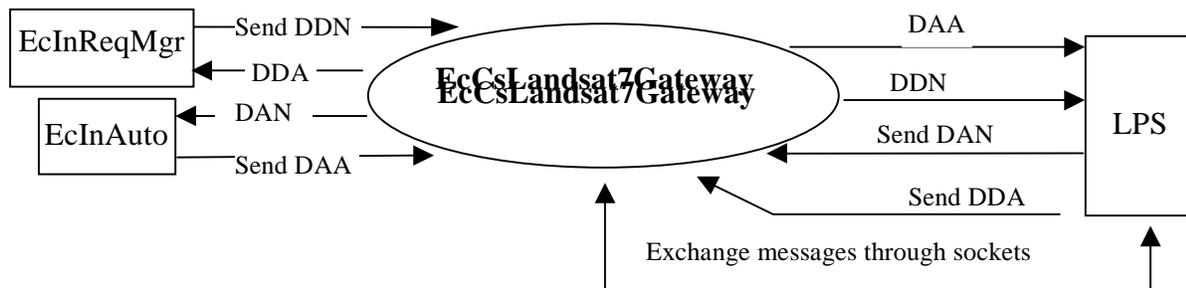


Figure 4.8.6.4.3-1. Landsat 7 Gateway Server Architecture Diagram

4.8.6.4.4 Landsat 7 Gateway Server Process Descriptions

Table 4.8.6.4.4-1 provides a description of the process shown in the Landsat 7 Gateway Server architecture diagram.

Table 4.8.6.4.4-1. Landsat 7 Gateway Server Process

Process	Type	COTS/ Developed	Description
EcCsLandsat7 Gateway	Server/ Client	Developed	<p>The Landsat 7 Gateway Server is the interface for automated data transfers between the LPS and the ECS. It enables LPS to send DANs and DDAs to the ECS and enables the ECS to send DAAs and DDNs to the LPS.</p> <p>The Landsat 7 Gateway Server provides two basic interfaces:</p> <ul style="list-style-type: none"> • Socket interface: used by the gateway and LPS to exchange messages. The gateway acts as both socket client and socket server. • RPC interface: used by the gateway and EclnAuto and EclnReqMgr to exchange messages. The gateway acts as both RPC client and RPC server. <p>The Landsat 7 Gateway Server supports:</p> <ul style="list-style-type: none"> • Multiple processes at a time • Asynchronous request processing.

4.8.6.4.5 Landsat 7 Gateway Server Process Interface Descriptions

Table 4.8.6.4.5-1 provides descriptions of the interface events shown in the Landsat 7 Gateway Server architecture diagram.

Table 4.8.6.4.5-1. Landsat 7 Gateway Server Process Interface Events (1 of 2)

Event	Event Frequency	Interface	Initiated by	Event Description
DAA	One per status exchange with EclnAuto	LPS	<i>Process:</i> EcCsLandsat7Gateway <i>Library:</i> RogueWaveNet Socket interface	The Data Availability Acknowledgment is forwarded by the EcCsLandsat7Gateway to the Landsat 7 Processing System (LPS) using sockets.
DDN	One per status exchange with EclnReqMgr	LPS	<i>Process:</i> EcCsLandsat7Gateway <i>Library:</i> RogueWaveNet Socket interface	The Data Delivery Notice is forwarded by the EcCsLandsat7Gateway to the Landsat 7 Processing System using sockets.
Send DAN	One per message exchange with LPS	<i>Process:</i> EcCsLandsat7G ateway <i>RogueWaveNet library</i> Socket interface	LPS	The LPS sends a DAN to the EcCsLandsat7Gateway when it has data for ingest.

Table 4.8.6.4.5-1. Landsat 7 Gateway Server Process Interface Events (2 of 2)

Event	Event Frequency	Interface	Initiated by	Event Description
Send DDA	One per message exchange with LPS	<i>Process:</i> EcCsLandsat7Gateway <i>RogueWaveNet library</i> Socket interface	LPS	The LPS sends a DDA to the EcCsLandsat7Gateway to acknowledge a delivery of ingested data.
Exchange messages through sockets	One per exchange with the LPS	<i>Process:</i> EcCsLandsat7Gateway <i>RogueWaveNet library</i> Socket Interface	LPS	The EcCsLandsat7Gateway sends and receives data status messages from the LPS through sockets.
Send DAA	One per status exchange with EclnAuto	<i>Process:</i> EcCsLandsat7Gateway <i>RogueWaveNet library</i> Socket interface	<i>Process:</i> EclnAuto <i>Class:</i> InAutoIngestIF_1_0_Mgr	The EclnAuto sends a DAA to the EcCsLandsat7Gateway to acknowledge the availability of data to be ingested.
DAN	One per message exchange with LPS	<i>Process:</i> EclnAuto <i>Class:</i> InAutoIngestIF_1_0_Mgr	<i>Process:</i> EcCsLandsat7Gateway <i>RogueWaveNet library</i> Socket interface	The EcCsLandsat7Gateway (using a RPC) forwards the Data Availability Notice to the EclnAuto.
DDA	One per message exchange with LPS	<i>Process:</i> EclnReqMgr <i>Class:</i> InRequest	<i>Process:</i> EcCsLandsat7Gateway <i>RogueWaveNet library</i> Socket interface	The Data Delivery Acknowledgment is forwarded to the EclnReqMgr by the EcCsLandsat7Gateway using a RPC.
Send DDN	One per status exchange with EclnReqMgr	<i>Process:</i> EcCsLandsat7Gateway <i>RogueWaveNet library</i> Socket interface	<i>Process:</i> EclnReqMgr <i>Class:</i> InRequest	The EclnReqMgr sends a DDN to the EcCsLandsat7Gateway to provide information about a delivery of ingested data.

4.8.6.4.6 Landsat 7 Gateway Server Data Stores

Data stores are not applicable for the Landsat 7 Gateway Server.

4.8.6.5 MOJO Gateway Server Computer Software Component Software Description

4.8.6.5.1 MOJO Gateway Server Functional Overview

The MOJO Gateway Server CSC provides a common interface and network address for all of the distributed ECS services accessible from the Java front end. The functionality provided by MOJO can be divided into four major groups:

- I. The MOJO Gateway Server provides session management, which can:
 - accept various request messages from Java Web Clients
 - maintain user state information and session information
 - verify users' session states
 - dispatch users' requests, such as advertisement search, subscriptions, submission/modification of DARs, DAR queries, and requests for user profiles via proxy objects
- II. The MOJO Gateway Server provides a security gateway to DCE/OODCE security services, which can:
 - login to DCE on behalf of a user
 - logout from DCE
- III. The MOJO Gateway Server provides a security gateway to various ECS services within MSS and IOS, which can:
 - spawn proxy objects to authenticated clients
 - do authenticated RPC to access ECS services on behalf of clients
- IV. The MOJO Gateway Server provides a gateway to Java Web Clients, which can:
 - send Java Web Clients the result messages from the ECS
 - flag the client if a request to the ECS has failed

4.8.6.5.2 MOJO Gateway Server Context

Figure 4.8.6.5.2-1 is the MOJO Gateway Server context diagram. Table 4.8.6.5.2-1 provides descriptions of the interface events in the MOJO Gateway Server context diagram.

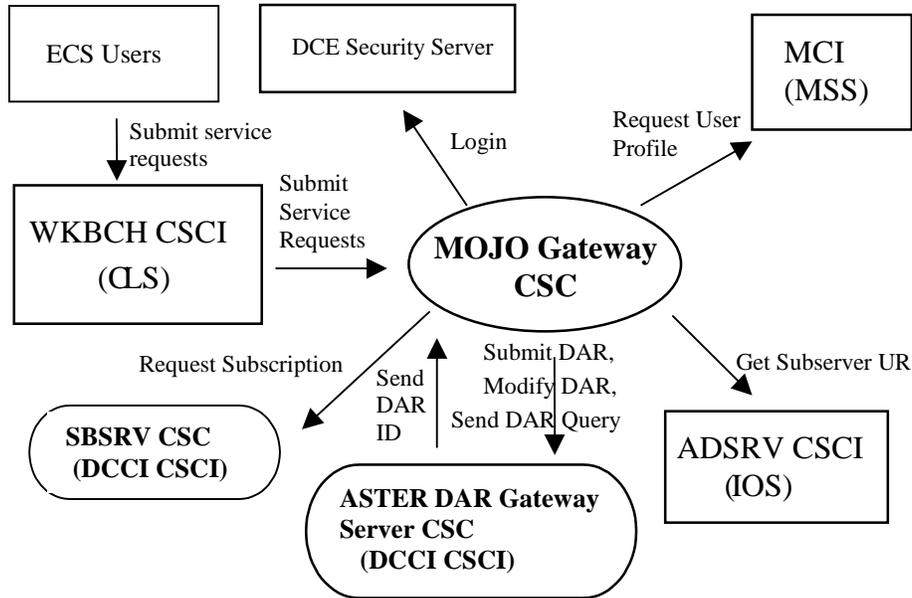


Figure 4.8.6.5.2-1. MOJO Gateway Server Context Diagram

Table 4.8.6.5.2-1. MOJO Gateway Server Interface Events (1 of 2)

Event	Interface Event Description
Login	The MOJO Gateway Server CSC logs into the Distributed Computing Environment (DCE) Security Server using the remote user's ID and password.
Request user profile	The MOJO Gateway Server CSC submits a request for a user profile to the MCI.
Get Subserver UR	The MOJO Gateway Server CSC submits a request to retrieve the correct subscription server UR from the ADSRV CSC (Advertising server).
Submit DAR	The MOJO Gateway Server CSC submits a Data Acquisition Request (DAR) to the ASTER DAR Gateway CSC. As the result of a DAR submission, the user receives a DAR ID (a string of characters used to track a DAR). The user receives notification every time data resulting from this DAR is received in the system.
Modify DAR	The MOJO Gateway Server CSC submits a modification to an existing DAR to the ASTER DAR Gateway CSC. As the result of a DAR submission, the user receives a different DAR ID. The user receives notification every time data resulting from this DAR is received in the system.
Send DAR Query	Submit a DAR query to the ASTER DAR Gateway. The query can be of type queryxARContents, queryxARSummary, or queryxARScenes. The query results are returned by the ASTER DAR Gateway.
Send DAR ID	The ASTER DAR Gateway Server extracts the returned DAR ID and sends it to the Java DAR Tool, via the MOJO Gateway Server.

Table 4.8.6.5.2-1. MOJO Gateway Server Interface Events (2 of 2)

Event	Interface Event Description
Request Subscription	The MOJO Gateway Server CSC submits requests to the SBSRV CSC for notification upon a specific event occurring in the system. An example is subscribing to the insert of a particular granule type. A valid subscription request results in the return of a subscription identifier. The subscription identifier is not returned to the user, but used by the MOJO Gateway Server CSC.
Submit Service Requests	The WKBCH CSCI submits a request for an ECS service on behalf of the user. Service types include Advertisement search, Subscriptions, User Profile updates, DAR submittal or modification.

4.8.6.5.3 MOJO Gateway Server Architecture

Figure 4.8.6.5.3-1 is the MOJO Gateway Server architecture diagram. The diagram shows the events sent to the MOJO Gateway Server process and the events the MOJO Gateway Server process sends to other processes.

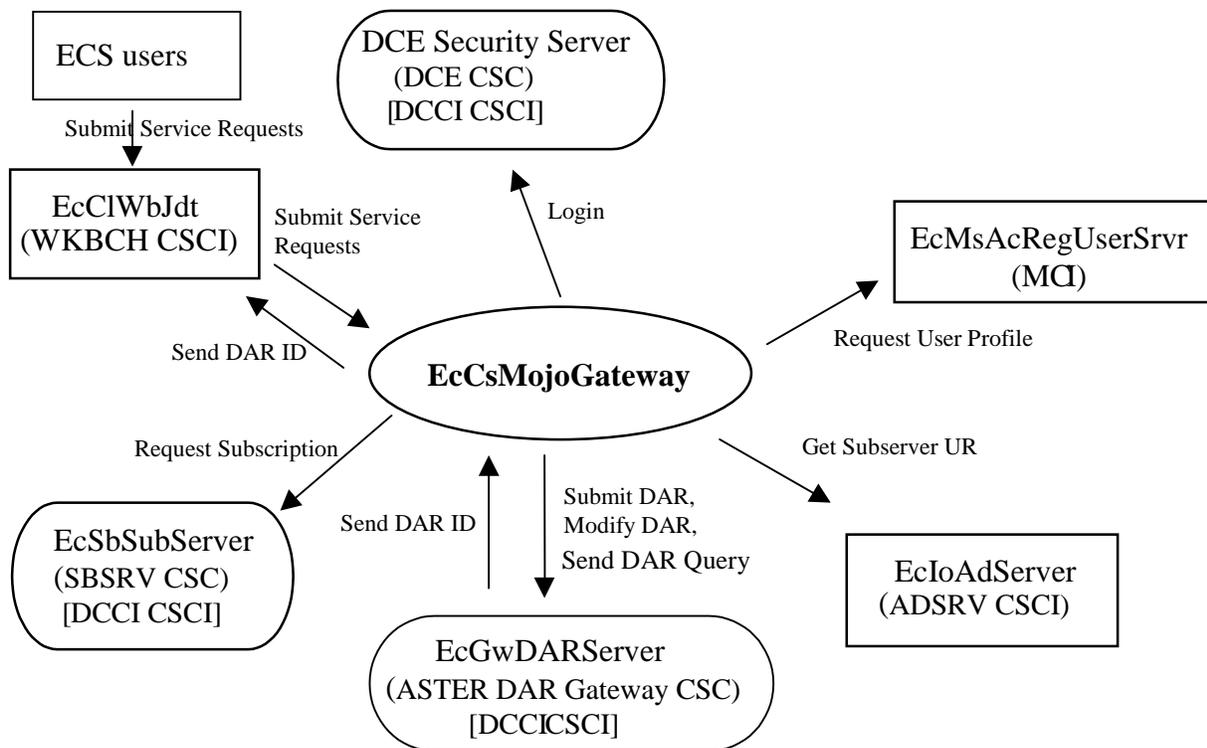


Figure 4.8.6.5.3-1. MOJO Gateway Server Architecture Diagram

4.8.6.5.4 MOJO Gateway Server Process Descriptions

Table 4.8.6.5.4-1 provides a description of the processes in the MOJO Gateway Server architecture diagram.

Table 4.8.6.5.4-1. MOJO Gateway Server Processes

Process	Type	COTS / Developed	Functionality
EcCsMojoGateway	Server	Developed	<p>The EcCsMojoGateway is a server that generally provides an internet gateway from JAVA WEB users to DCE/OODCE based ECS services. It provides a session management for Java Web users, which can accept various requests and maintain user's information and session information. It also provides an interface to the DCE security service, which can authenticate Java WEB users. It also provides another interface to various ECS services, such as Science Data service, for Java WEB users to search and retrieve earth science data. Finally, it provides an interface to JESS, which can send various messages back to Java WEB users.</p> <p>As a server developed by ECS, it provides the following major interface functionality:</p> <ul style="list-style-type: none"> • MjGetJava WebMessage: Reads a message from JESS • MjProcessRequest: This is a polymorphic method, which defines a common interface and by which the various DCE/OODCE and ECS service facilities are invoked • MjSendJava WebMessage: Sends a message to JESS <p>The EcCsMojoGateway supports:</p> <ul style="list-style-type: none"> • Multiple concurrent requests • Request processing de-coupled from rpc thread and UNIX process

4.8.6.5.5 MOJO Gateway Server Process Interface Descriptions

Table 4.8.6.5.5-1 provides descriptions of the interface events shown in the MOJO Gateway Server architecture diagram.

Table 4.8.6.5.5-1. MOJO Gateway Server Process Interface Events (1 of 3)

Event	Event Frequency	Interface	Initiated by	Event Description
Login	One per user session	<i>Process:</i> DCE Security Server (COTS) <i>Library:</i> EcSeLoginContext <i>Class:</i> DCEStdLoginContext	<i>Process:</i> EcCsMojoGateway <i>Library:</i> EcCsMojoGateway <i>Class:</i> EcMjDCELoginProxy	The EcCsMojoGateway sends a request to log into the DCE Security Server using the remote user's ID and password.
Request User Profile	Per user request	<i>Process:</i> EcMsAcRegUserSrvr <i>Library:</i> MsAcCInt, MsAcComm <i>Class:</i> EcAcProfileMgr	<i>Process:</i> EcCsMojoGateway <i>Library:</i> EcCsMojoGateway <i>Class:</i> EcMjRetrieveProfileProxy	The EcCsMojoGateway submits a request for a user profile to the EcMsAcRegUserSrvr.
Get Subserver UR	One per request	<i>Process:</i> EcIoAdServer <i>Library:</i> IoAdvSearch <i>Class:</i> EcIoAdSearch	<i>Process:</i> EcCsMojoGateway <i>Library:</i> EcCsMojoGateway <i>Class:</i> EcMjEcsAdsrvProxy	The EcCsMojoGateway submits a request to retrieve the correct subscription server UR from the EcIoAdServer.
Submit DAR	Per user request	<i>Process:</i> EcGwDARServer <i>Library:</i> EcGwDAR <i>Class:</i> EcGwDAR_SubmitDataRequest_C	<i>Process:</i> EcCsMojoGateway <i>Library:</i> EcCsMojoGateway <i>Class:</i> EcMjDarSubmitDarProxy	The EcCsMojoGateway submits a Data Acquisition Request (DAR) to the EcGwDARServer. As the result of a DAR submission, the user receives a DAR ID (a string of characters used to track a DAR). The user receives notification every time data resulting from this DAR is received in the system.

Table 4.8.6.5.5-1. MOJO Gateway Server Process Interface Events (2 of 3)

Event	Event Frequency	Interface	Initiated by	Event Description
Modify DAR	Per user request	<i>Process:</i> EcGwDARServer <i>Library:</i> EcGwDAR <i>Class:</i> EcGwDAR_ModifyDarRequest_C	<i>Process:</i> EcCsMojoGateway <i>Library:</i> EcCsMojoGateway <i>Class:</i> EcMjDarModifyDarProxy	The EcCsMojoGateway submits a modification to an existing DAR to the EcGwDARServer. As the result of a DAR submission, the user receives a different DAR ID. The user receives notification every time data resulting from this DAR is received in the system.
Send DAR Query	Per user request	<i>Process:</i> EcGwDARServer <i>Library:</i> EcGwDAR <i>Class:</i> EcGwDARQueryxARContentsRequest_C, EcGwDARQueryxARSummaryRequest_C, EcGwDARQueryxARScenesRequest_C	<i>Process:</i> EcCsMojoGateway <i>Library:</i> EcCsMojoGateway <i>Classes:</i> EcMjDarQueryxARContentsProxy, EcMjDarQueryxARSummaryProxy, EcMjDarQueryxARScenesProxy	The EcCsMojoGateway sends a query request to the EcGwDARServer. The EcGwDARServer returns the query results.
Send DAR ID	One per DAR ID sent	<i>Process:</i> EcCsMojoGateway <i>Library:</i> EcGwDAR <i>Class:</i> EcGwDARSubmitDarRequest_C	<i>Process:</i> EcGwDARServer <i>Class:</i> EcGwDARGatewayRequest_S	The EcGwDARServer extracts the returned DAR ID and sends it to the Java DAR Tool, via the EcCsMojoGateway.
Request Subscription	Per user request	<i>Process:</i> EcSbSubServer <i>Library:</i> EcSbCI <i>Class:</i> EcCISubscription	<i>Process:</i> EcCsMojoGateway <i>Library:</i> EcCsMojoGateway <i>Class:</i> EcMjECSSbsrvProxy	A request for notification upon a specific event occurring in the system is sent to the EcSbSubServer. A valid subscription request results in the return of a subscription identifier. The subscription identifier is not returned to the user, but used by the EcCsMojoGateway.

Table 4.8.6.5.5-1. MOJO Gateway Server Process Interface Events (3 of 3)

Event	Event Frequency	Interface	Initiated by	Event Description
Submit Service Requests	One type per user request	<i>Process:</i> EcCsMojoGateway <i>Library:</i> EcCsMojoGateway <i>Class:</i> EcMjECSSbsrvProxy	<i>Process:</i> EcCIWbJdt <i>Library:</i> Standard JDK 1.1.x socket support <i>Class:</i> Java.net.Socket	The EcCIWbJdt submits a request for an ECS service on behalf of users to the EcCsMojoGateway. The only service type available is the Subscription request.

4.8.6.5.6 MOJO Gateway Server Data Stores

Data stores are not applicable for the MOJO Gateway Server.

4.8.6.6 Configuration Registry Server Software Description

4.8.6.6.1 Configuration Registry Server Functional Overview

The Configuration Registry Server provides a single interface to retrieve configuration attribute-value pairs for ECS Servers from the Configuration Registry Database, via a Sybase Server. The Configuration Registry Server maintains an internal representation of the tree in which configuration attribute-value pairs are stored. General configuration parameters used by many servers are stored in higher nodes in the tree. Parameters specific to a single ECS Server are contained in the leaf nodes of the tree.

The Configuration Registry Server accepts queries to the Configuration Registry Database with a configuration path and returns a list of attribute-value pairs. A wild-card character may be specified as the last element in the path to retrieve all attributes in the sub-tree specified. Each Configuration Registry Server is MODE specific, with multiple Registry Servers running in a mode to provide redundancy.

4.8.6.6.2 Configuration Registry Server Context

Figure 4.8.6.6.2-1 is the Configuration Registry Server context diagram. Table 4.8.6.6.2-1 provides descriptions of the interface events in the Configuration Registry Server context diagram.

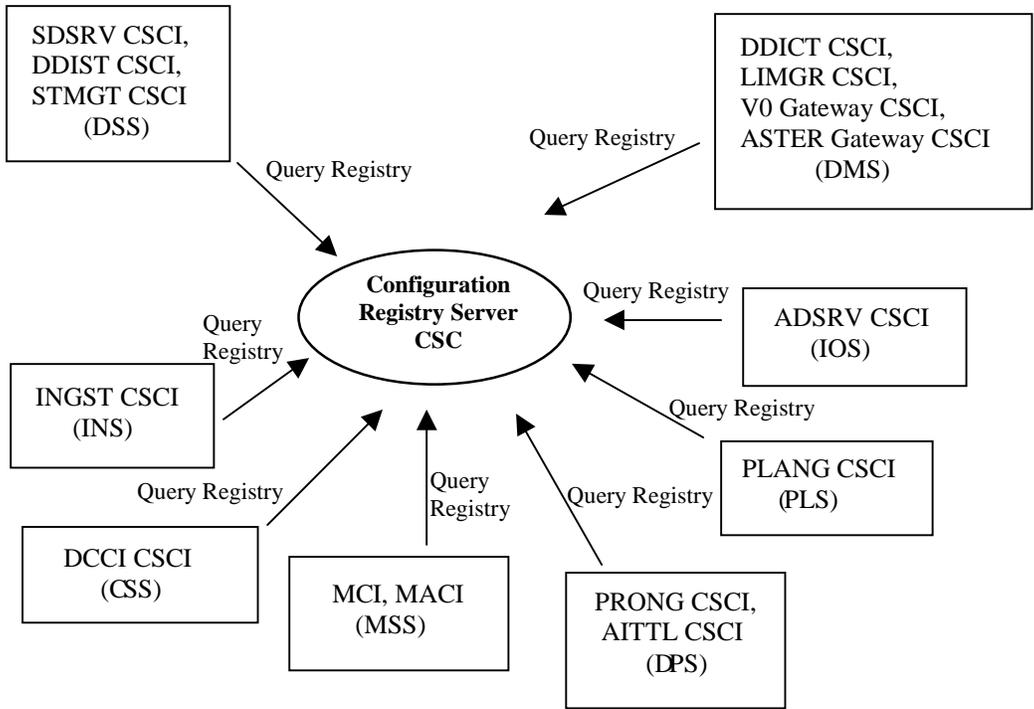


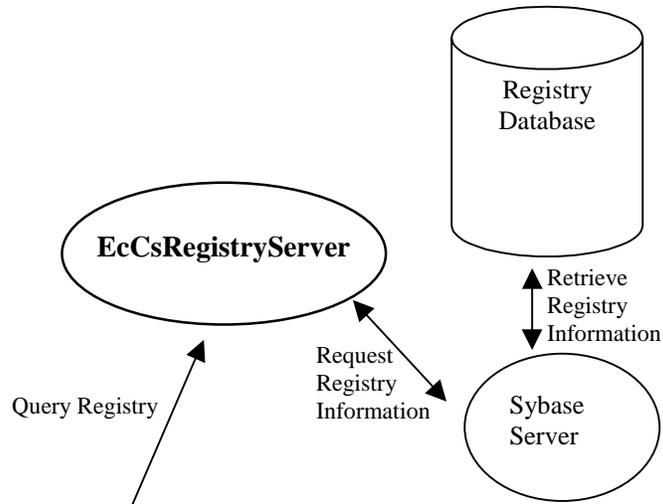
Figure 4.8.6.6.2-1. Configuration Registry Server Context Diagram

Table 4.8.6.6.2-1. Configuration Registry Server Interface Events

Event	Interface Event Description
Query Registry	Upon startup, ECS Servers query the Configuration Registry Server for configuration parameters and their respective value(s). The ECS Servers pass in a path that corresponds to a sub-tree in the MODE configuration value tree maintained by the server. The Registry Server uses this path as a starting point in the tree and returns all parameters and their associated values in the sub-tree below it.

4.8.6.6.3 Configuration Registry Server Architecture

Figure 4.8.6.6.3-1 is the Configuration Registry Server architecture diagram. The diagram shows the events sent to the Configuration Registry Server process and the events the Configuration Registry Server process sends to other processes.



ECS Applications –EcDsScienceDataServer, EcDsHdfEosServer, EcDsDistributionServer, EcDsStArchiveServer, EcDsStStagingMonitorServer, EcDsStStagingDiskServer, EcDsSt8MMServer, EcDsStD3Server, EcDsStIngestFtpServer, EcDsStFtpDisServer, EcDsStPrintServer, EcDsStPullMonitorServer, EcInAuto, EcInPolling, EcInReqMgr, EcInMailGWServer, EcInGran, EcCIWbJdt, EcCIDtUserProfileGateway, EcDmDictServer, EcDmDdMaintenanceTool, EcDmLimServer, EcDmV0ToEcsGateway, EcDmEcsToV0Gateway, EcDmAsterToEcsGateway, EcDmEcsToAsterGateway, EcIoAdServer, EcPISubMgr, EcPIOnDemandMgr, EcDpPrDeletion, EcDpPrJobMgmt, EcDpPrJobMgmtClient, EcSbSubServer, EcGwDARServer, EcCsEmailParser, EcCsLandsat7Gateway, EcCsMojoGateway, EcMsAcRegUserSrvr, EcMsAcOrderSrvr, EcCIWbFoliodProxyServer and EcCLWbJessProxyServer.

Figure 4.8.6.6.3-1. Configuration Registry Server Architecture Diagram

4.8.6.6.4 Configuration Registry Server Process Descriptions

Table 4.8.6.6.4-1 provides a description of the processes in the Configuration Registry Server architecture diagram.

Table 4.8.6.6.4-1. Configuration Registry Server Processes

Process	Type	COTS / Developed	Functionality
EcCsRegistryServer	Server	Developed	The Configuration Registry Server provides a single interface to retrieve configuration attribute-value pairs for ECS Servers from the Configuration Registry Database, via a Sybase Server. The Configuration Registry Server accepts queries to the Configuration Registry Database with a configuration path and returns a list of attribute-value pairs. A wild-card character may be specified as the last element in the path to retrieve all attributes in the sub-tree specified.

4.8.6.6.5 Configuration Registry Server Process Interface Descriptions

Table 4.8.6.6.5-1 provides descriptions of the interface events shown in the Configuration Registry Server architecture diagram.

Table 4.8.6.6.5-1. Configuration Registry Server Process Interface Events

Event	Event Frequency	Interface	Initiated by	Event Description
Retrieve Registry Information	One or more per client request	Registry Database	Sybase Server (COTS)	The Sybase Server receives the request and retrieves the necessary attribute-value pairs and returns them to the EcCsRegistryServer.
Request Registry Information	One or more per client request	Sybase Server (COTS)	<i>Process:</i> EcCsRegistryServer <i>Class:</i> EcRgRegistryServer_S	The Configuration Server sends the request to the Sybase Server to retrieve the attribute-value pairs.
Query Registry	One per client request	<i>Process:</i> EcCsRegistryServer <i>Library:</i> EcCsRegistryClient <i>Class:</i> EcRgRegistryServer_C	<i>Processes:</i> All ECS applications (identified in the architecture diagram)	An ECS application sends a query request to the Configuration Server to retrieve a list of attribute-value pairs needed by the application.

4.8.6.6.6 Configuration Registry Server Data Stores

The Configuration Registry Server uses a Sybase Server database for its persistent storage. The following is a brief description of the types of data contained in the database:

- **Mode:** This data store contains the list of modes and a description of the purpose of the mode.
- **Node:** This data store contains information that describes each node in the tree.
- **NodeContact:** This data store contains the information of the person who is responsible for the information contained in each node of the tree.
- **Attribute tree:** This data store contains a list of tree names and a description of each tree.
- **Attribute:** This data store contains a description of each attribute whose value is assigned to a particular node.
- **AttributeValidEnum:** This data store contains enumerated string values for attributes of enumerated types.
- **AccessControlList:** This data store contains the access control information for each node.

- **ConfiguredValue:** This data store contains the value for the parameter stored in a node, and associated information.
- **ConfigurationManagementContact:** This data store contains a list of configuration management contacts for information stored in the Configuration Registry.

Table 4.8.6.6-1 provides descriptions of the data found in the separate Sybase data stores used by the Configuration Registry Server. More detailed information on these data stores can be found in the Configuration Registry Database Design and Schema Specifications for the ECS Project (Refer to CDRL 311).

Table 4.8.6.6-1. Configuration Registry Server Data Stores

Data Store	Type	Functionality
Mode	Sybase	This data store contains the list of modes and a description of the purpose of the mode. It also contains a mapping of the mode to the tree name.
AttributeTree	Sybase	This data store contains a list of tree names and a description of each tree.
Node	Sybase	This data store contains information that describes each node in the tree. This information includes a NodeID, the tree name to which it belongs, the node name, its parent NodeID, the node type, and a node description.
NodeContact	Sybase	This data store contains the information of the person who is responsible for the information contained in each node of the tree. It includes the NodeID, and the FirstName, LastName, Org(anization), and Email address of the person responsible.
AccessControlList	Sybase	This data store contains the access control information for each node. It includes the NodeID, an AclSequenceNumber, AclType, AclUser, AclGroup, and Create, Read, Update, and Delete flags.
Attribute	Sybase	This data store contains a description of each attribute whose value is assigned to a particular node. It lists the attribute type, minimum and maximum values, and the NodeID.
AttributeValidEnum	Sybase	This data store contains enumerated string values for attributes of enumerated type. It includes a string name for each enumerated value, a description of the value, and a NodeID.
ConfiguredValue	Sybase	This data store contains the value for the parameter stored in a node, and associated information. It includes the NodeID, DataType, and TimeStamp of last change, Comment for the change, Float, Integer, or String value, ValueVersion number, and userid of the user who made the change.
ConfigurationManagementContact	Sybase	This data store contains a list of configuration management contacts for information stored in the Configuration Registry.

4.8.6.7 Distributed Computing Environment Support Group Description

The Distributed Computing Environment (DCE) support group consists of the Security Authentication service, the Time Service, and the Server Locator functions.

4.8.6.7.1 Distributed Computing Environment Functional Overview

The Security Authentication of the CSS Security Service enables the server processes to obtain a valid login context for DCE. All servers are required to use Security Authentication to login to DCE and perform the DCE operations for normal execution.

The CSS Time Service uses the DCE Distributed Time Service (DTS) to synchronize the system clocks on the ECS hosts by directly adjusting the operating system time on each host as needed. When a host clock needs to be advanced, the time adjustment is made in transparent increments until the correction is complete. When a host clock is found to be ahead of the actual time, the host clock is slowed down transparently in increments until the correction is complete. (A host clock is never adjusted in a backward direction). The CSS Time Service provides time within a millisecond resolution. ECS applications use the APIs provided by the CSS Time Service to obtain time in various formats. Applications needing to simulate a time other than the current time apply a specified delta time to the current time. Time classes enable applications to obtain the current time in various formats with or without a predetermined delta time.

The Server Locator of the CSS enables clients to locate and communicate with the various ECS servers. The ECS servers register their location information into the Cell Directory Service (CDS) of the Server Locator independent of the server's physical location. Servers registering in the CDS are available to be accessed by other application clients. Clients use the Server Locator and the ECS operating mode to find the server of interest.

4.8.6.7.2 Distributed Computing Environment Context

Figure 4.8.6.7.2-1 is the Distributed Computing Environment context diagram. Table 4.8.6.7.2-1 provides descriptions of the interface events shown in the Distributed Computing Environment context diagram.

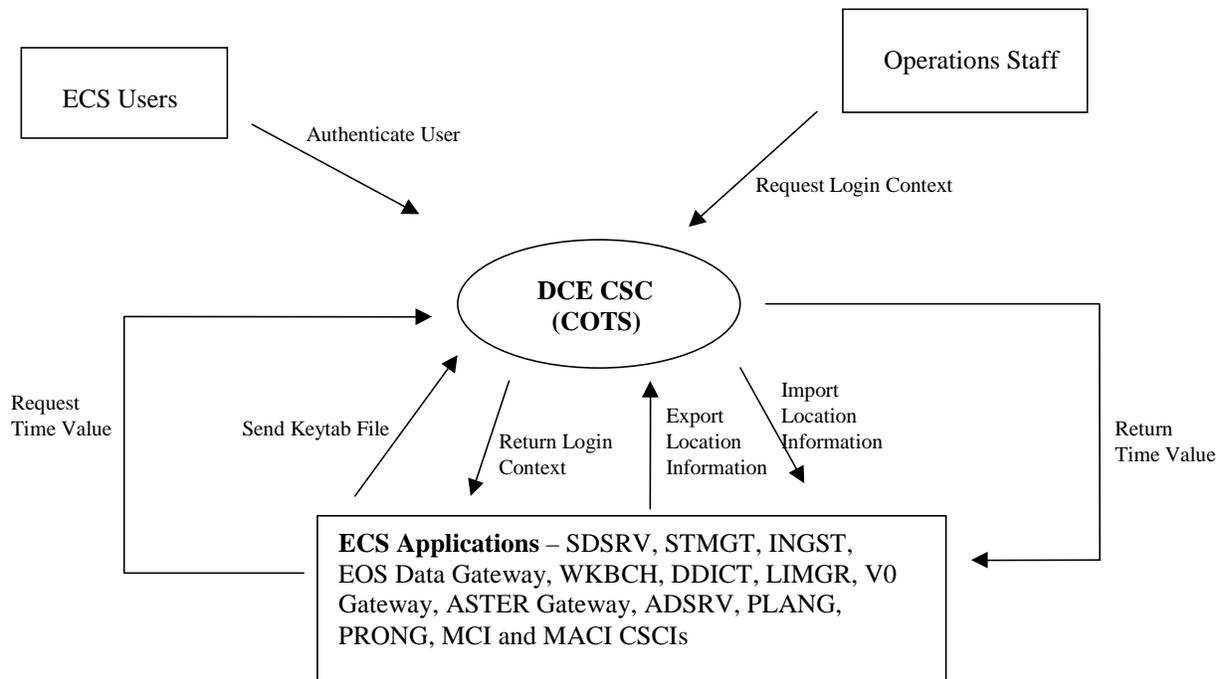


Figure 4.8.6.7.2-1. Distributed Computing Environment Context Diagram

Table 4.8.6.7.2-1. Distributed Computing Environment Interface Events

Event	Interface Event Description
Request Login Context	The Operations staff can request a login context via a GUI.
Return Time Value	The DCE CSC returns a time value to the applications.
Import location information (binding information)	An ECS application retrieves server location information from CDS via the Server Locator.
Export location information (binding information)	The DCE CSC places physical and logical location information in CDS via the Server Locator.
Return Login Context	The DCE CSC creates a login context on behalf of the DCE Security server. The login context is returned to the ECS application upon request.
Send Keytab File	ECS Applications provides a keytab file to the DCE CSC to obtain a login key (password).
Request Time Value	Applications submit time requests to the DCE CSC.
Authenticate User	ECS users send user id and password information, via the WKBCH CSCI and the V0 Client, to the DCE CSC. In response, the DCE CSC returns an authentication status to the user.

4.8.6.7.3 Distributed Computing Environment Architecture

Figure 4.8.6.7.3-1 is the Distributed Computing Environment (DCE) support group architecture diagram. The diagram shows the events sent to the DCE processes and the events the DCE processes send to other processes.

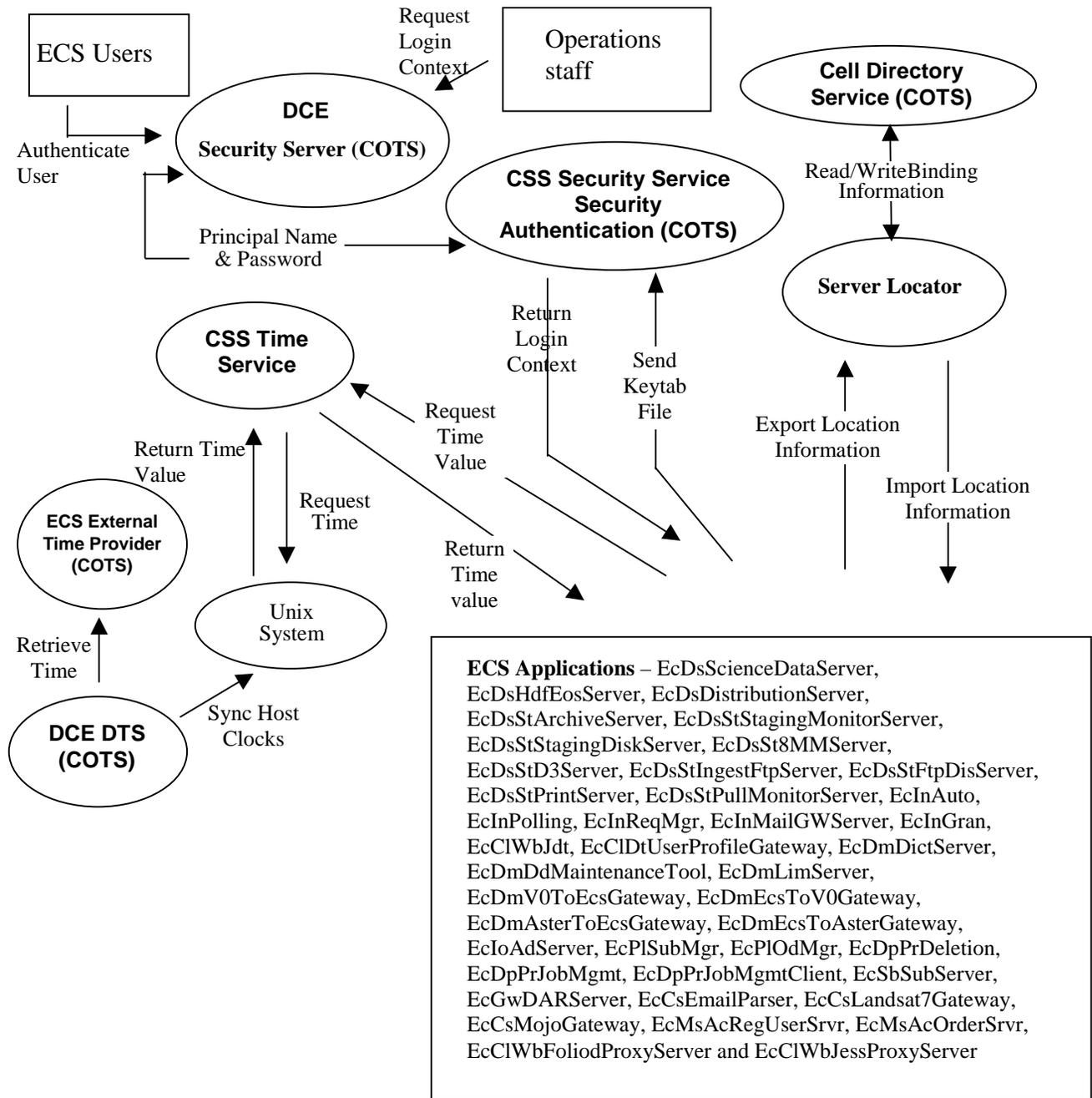


Figure 4.8.6.7.3-1. Distributed Computing Environment Architecture Diagram

4.8.6.7.4 Distributed Computing Environment Process Descriptions

Table 4.8.6.7.4-1 provides descriptions of the processes shown in the Distributed Computing Environment architecture diagram. ECS applications provide Keytab files to Security Authentication and receives the returned Login context. Provides export location information to the Server locator. Requests Time from the Time Service and receives a Time value as the result. Provides Host clocks information to Distributed Time Service and receives Sync Host Clocks information for clock synchronization. Uses the Server Locator to retrieve server location information from CDS.

Table 4.8.6.7.4-1. Distributed Computing Environment Processes

Process	Type	COTS/ Developed	Functionality
DCE Security Server	Server	COTS	Stores registry information and verifies login names and passwords.
CSS Security Service Security Authentication	Server	COTS	Receives Keytab files and returns a Login context for valid users. Receives and returns Principal Name and Password information to the Security Server for login validation.
Server Locator	Server	Developed	Stores and retrieves server location data on the CDS.
Cell Directory Service	Internal	COTS	Stores server location information and provides interfaces for storing and retrieving the location information.
DCE DTS	Server	COTS	Receives the delta time from an external time from the CSS Time Service and adjusts the UNIX clock as needed.
CSS Time Service	Server	Developed	Retrieves the current time from an external time provider and provides precise time in server requested formats
ECS External Time Provider	Server	COTS	Provides accurate time to synchronize the DCE cell.

4.8.6.7.5 Distributed Computing Environment Process Interface Descriptions

Table 4.8.6.7.5-1 provides descriptions of the interface events shown in the Distributed Computing Environment architecture diagram.

**Table 4.8.6.7.5-1. Distributed Computing Environment Process Interface Events
(1 of 3)**

Event	Event Frequency	Interface	Initiated By	Event Description
Read/Write Binding Information	A few times per server startup and per servers lookup	Process: Cell Directory Service (CDS) [COTS] Library: PF Class: EcPfManagedServer	Process: Server Locator	The Server Locator communicates with CDS to read and write server location information.
Import location information (binding information)	One per server	Processes: ECS Applications (All servers identified in the architecture diagram.)	Process: Server Locator Library: PF Class: EcPfManagedServer	An ECS application retrieves server location information from CDS via the Server Locator.
Export location information (binding information)	One per server	Process: Server Locator Library: PF Class: EcPfManagedServer	Processes: ECS Applications (All processes identified in the architecture diagram.)	The ECS application places physical and logical location information in CDS via the Server Locator.
Send Keytab File	One per application	Process: CSS Security Service Security Authentication (COTS) Library: EcSeServerKeyMgmt Class: EcSeServerKeyMgmt	Processes: ECS Applications (All processes identified in the architecture diagram.)	ECS applications provide a keytab file to the CSS Security Service Security Authentication to obtain a login key (password).
Return Login Context	One per user	ECS Applications (All servers identified in the architecture diagram.)	Process: DCE Security Server Library: EcSeLoginContext Class: EcSeLoginContext	The DCE Security Server creates a login context on behalf of the user. The login context is returned to the application upon request.

**Table 4.8.6.7.5-1. Distributed Computing Environment Process Interface Events
(2 of 3)**

Event	Event Frequency	Interface	Initiated By	Event Description
Request Time Value	One per request	<i>Process:</i> CSS Time Service (COTS) <i>Library:</i> EcTiTime <i>Class:</i> EcTiTimeService	<i>Processes:</i> ECS Applications (All processes identified in the architecture diagram.)	Applications submit time requests to the CSS Time Service.
Return Time Value	One per request	<i>Process:</i> CSS Time Service (COTS) <i>Class:</i> EcTiTimeService	<i>Processes:</i> ECS Applications (All processes identified in the architecture diagram.)	The CSS Time Service returns a time value to the applications.
Request time	One per time request	Unix Operating System	<i>Process:</i> CSS Time Service <i>Class:</i> EcTiTimeService	The CSS Time Service submits requests to the Unix Operating System for current time values.
Sync Host Clocks	As needed	Unix Operating System	<i>Process:</i> DCE DTS	DCE DTS adjusts the host's operating system clock, as needed to maintain host clock synchronization.
Retrieve Time	One per time request	<i>Process:</i> ECS external time provider (COTS)	<i>Process:</i> DCE DTS <i>Class:</i> EcTiTimeService	The DCE DTS retrieves a timestamp from the ECS external time provider.
Return time Value	One per time request	<i>Process:</i> CSS Time Service <i>Class:</i> EcTiTimeService	Unix Operating System	The Unix Operating System returns time values to the CSS Time Service.
Principal Name & Password	One per request per server	<i>Process:</i> CSS Security Service Security Authentication (COTS)	<i>Process:</i> DCE Security Server (COTS) <i>Library:</i> EcSeLoginContext <i>Class:</i> EcSeLoginContext	The CSS Security Service Security Authentication process communicates with the DCE Security Server process to verify the server's principal login name and password.

**Table 4.8.6.7.5-1. Distributed Computing Environment Process Interface Events
(3 of 3)**

Event	Event Frequency	Interface	Initiated By	Event Description
Authenticate User	One per user	<i>Process:</i> DCE Security Server (COTS) <i>Libraries:</i> EcSeLogin, EcSeLoginContext <i>Classes:</i> EcSeLogin, EcSeLoginContext	User <i>Processes:</i> EcCIWbJdt <i>Class:</i> UserProfileManager	ECS users send user id information to the EcCIWbJdt via the Web Browser. The EcCIWbJdt submits the user id information to the EcCsMojoGateway CSC. The EcCsMojoGateway CSC sends a request to the DCE Security Server to authenticate the user.
Request Login Context	One per operator request	<i>Process:</i> DCE Security Server (COTS) <i>Libraries:</i> EcSeLogin, EcSeLoginContext <i>Classes:</i> EcSeLogin, EcSeLoginContext	Operations Staff <i>Process:</i> Dce_login	The Operations staff can request a login context from the DCE Security Server via a GUI.

4.8.6.7.6 Distributed Computing Environment Data Stores

Table 4.8.6.7.6-1 provides a description of the data store shown in the Distributed Computing Environment architecture diagram.

Table 4.8.6.7.6-1. Distributed Computing Environment Data Stores

Data Store	Type	Functionality
ServerUR.map	Other	A flat file for the Server Locator classes to map short, logical service names to CDS entry names.

4.8.6.8 Remote File Access Group - File Transfer Protocol Description

The remote file access group consists of five functional processes described separately. The five processes are File Transfer Protocol (FTP), File Transfer Protocol Notification (FTPN), Network File System (NFS), Bulk Data Server (BDS), and Filecopy.

4.8.6.8.1 File Transfer Protocol Functional Overview

FTP is a user interface to the Internet standard File Transfer Protocol. With FTP a user is able to transfer files to and from remote network sites. FTP is client-server software where the user starts the client program first while the FTP daemon is the server started on the target machine.

4.8.6.8.2 File Transfer Protocol Context

Figure 4.8.6.8.2-1 is the FTP context diagram. Table 4.8.6.8.2-1 provides descriptions of the interface events in the FTP context diagram.

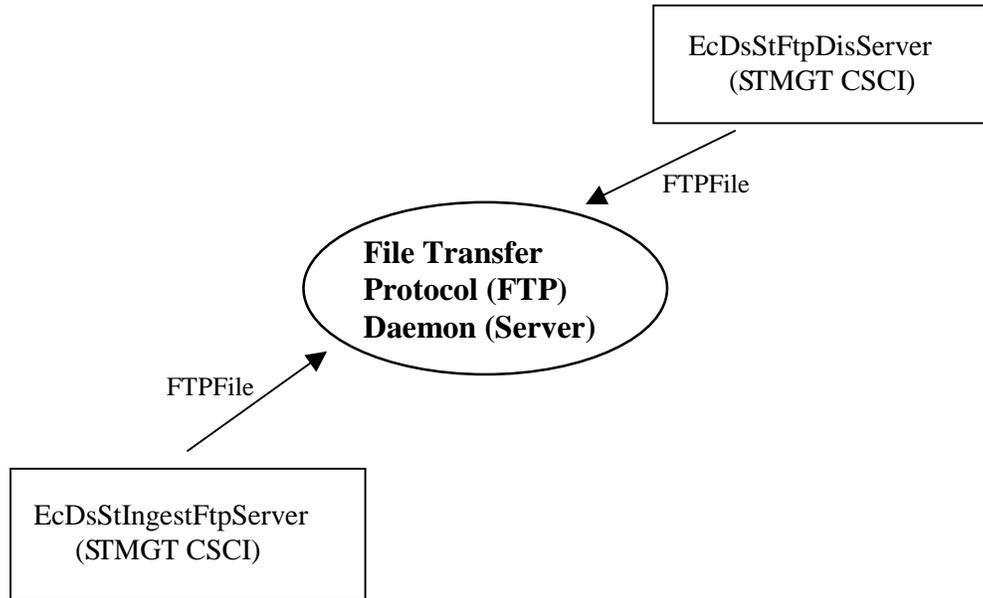


Figure 4.8.6.8.2-1. File Transfer Protocol Server Context Diagram

Table 4.8.6.8.2-1. File Transfer Protocol Interface Events

Event	Interface Event Description
FTPFile	The EcDsStIngestFtpServer sends requests to the FTP Daemon to transfer the files to the specified destination. The EcDsStFtpDisServer sends requests to the FTP Daemon to transfer files to the Pull cache or to an external user.

4.8.6.8.3 File Transfer Protocol Architecture

Figure 4.8.6.8.3-1 is the File Transfer Protocol architecture diagram. The ECS FTP is the standard UNIX utility with the CSS wrapper classes applied to provide additional ECS-developed capabilities. The CSS wrapper classes also provide APIs for more control and easier access to other applications.

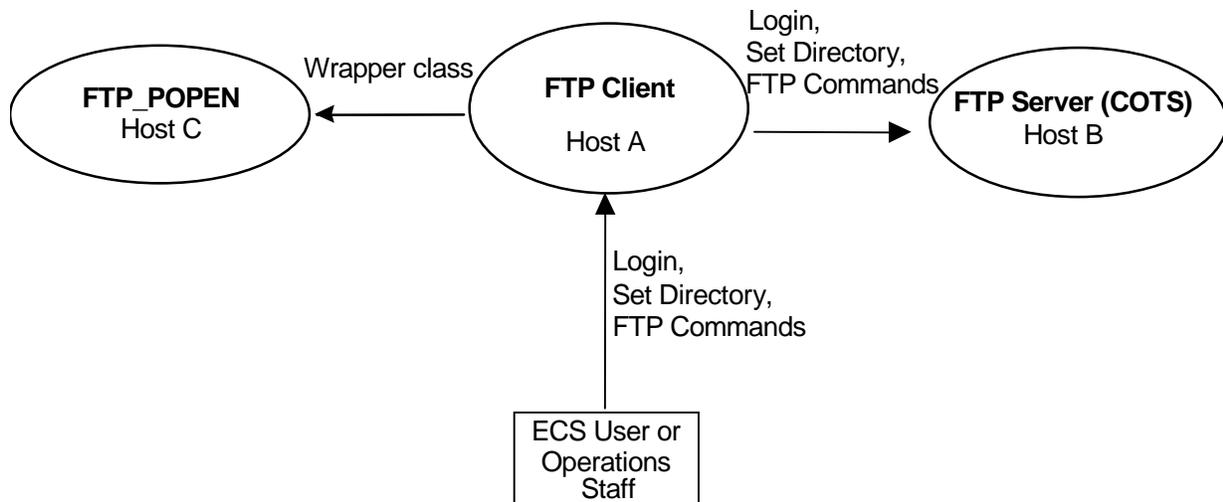


Figure 4.8.6.8.3-1. File Transfer Protocol Architecture Diagram

4.8.6.8.4 File Transfer Protocol Process Descriptions

Table 4.8.6.8.4-1 provides descriptions of the processes shown in the File Transfer Protocol architecture diagram.

Table 4.8.6.8.4-1. File Transfer Protocol Processes

Process	Type	COTS/ Developed	Functionality
FTP Server	Server	COTS	Provides basic FTP capabilities.
FTP Client w/ ECS FTP	API	Developed	Provides ECS specific additions for improved access by ECS applications to FTP servers.
User Process (FTP_POOPEN)	Client Application	Developed	ECS application processes use the Wrapper class whenever FTP is used with other class methods.

4.8.6.8.5 File Transfer Protocol Process Interface Descriptions

Table 4.8.6.8.5-1 provides descriptions of the interface events shown in the File Transfer Protocol architecture diagram.

Table 4.8.6.8.5-1. File Transfer Protocol Process Interface Events

Event	Event Frequency	Interface	Initiated by	Event Description
Login	One per connection to FTP	<i>Process:</i> FTP Server Daemon <i>Library:</i> CsFtFTPRelA <i>Class:</i> CsFtFTPRelA	User/Operations Staff <i>Process:</i> FTP Client <i>Classes:</i> DsStFtpWrappers, EcMjECSFtpProxy, MsMdHost, InMediaIngestRPUtil, InMessage, DpPrDataManager	The FTP Client, on behalf of a user or Operations Staff member, establishes the connection with the File Transfer Protocol Daemon.
Set Directory	One per directory set	<i>Process:</i> FTP Server <i>Library:</i> CsFtFTPRelA <i>Class:</i> CsFtFTPRelA	User/Operations Staff <i>Process:</i> FTP Client <i>Classes:</i> DsStFtpWrappers, EcMjECSFtpProxy, MsMdHost, InMediaIngestRPUtil, InMessage, DpPrDataManager	A User or Operations staff member sends commands, to the FTP Client, for setting the (working) directory (on the client and server).
FTP Commands	One per file transfer	<i>Process:</i> FTP Server <i>Library:</i> CsFtFTPRelA <i>Class:</i> CsFtFTPRelA	User/Operations Staff <i>Process:</i> FTP Client <i>Classes:</i> DsStFtpWrappers, EcMjECSFtpProxy, MsMdHost, InMediaIngestRPUtil, InMessage, DpPrDataManager	A User or Operations staff member sends commands, to the FTP Client, to transfer files from host to host. The commands are submitted to the FTP Server.
Wrapper class	One per FTP	<i>Process:</i> FTP_POPEN <i>Library:</i> CsFtFTPRelA <i>Class:</i> CsFtFTPRelA	<i>Process:</i> FTP Client <i>Classes:</i> DsStFtpWrappers, EcMjECSFtpProxy, MsMdHost, InMediaIngestRPUtil, InMessage, DpPrDataManager	The FTP Client provides wrapper functions, to the FTP_POPEN, to carry out FTP between two hosts.

4.8.6.8.6 File Transfer Protocol Data Stores

Data stores are not applicable for the File Transfer Protocol service.

4.8.6.9 Remote File Access Group - File Transfer Protocol Notification

4.8.6.9.1 File Transfer Protocol Notification Functional Overview

The CSS provides an FTP Notification to the Pull Monitor task (in the DSS STMGT) upon completion of FTP pulls from the pull disk area of the DSS STMGT. The CsFtFTPNotify Class provides a method invoked by the Pull Monitor at predefined time intervals. The CsFtFTPNotify extracts the successful FTP information from the SYSLOG file FTPD Debug messages and sends the involved directory and file names to the Pull Monitor.

4.8.6.9.2 File Transfer Protocol Notification Context

Figure 4.8.6.9.2-1 is the File Transfer Protocol Notification context diagram. Table 4.8.6.9.2-1 provides descriptions of the interface events shown in the File Transfer Protocol Notification context diagram.

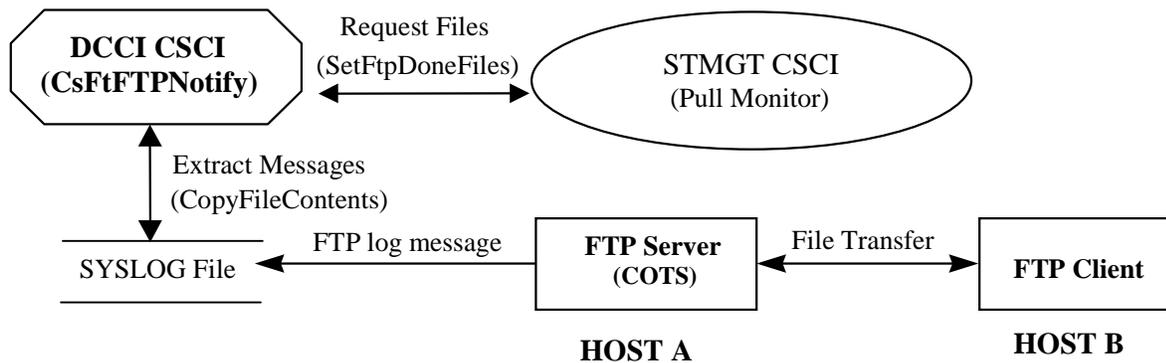


Figure 4.8.6.9.2-1. File Transfer Protocol Notification Context Diagram

Table 4.8.6.9.2-1. File Transfer Protocol Notification Interface Events

Event	Interface Event Description
File Transfer	The file transfer of specified files between the client and server.
FTP log message	The FTP server logs all FTP events to the system log (syslog) file.
Extract messages	The "CopyFileContents" f application copies the log files.
Request Files	An ECS application (Pull Monitor) requests a transfer of files via the FTP service by the SetFtpDone function. After the transfer, FTPNotify extracts information from the syslog file and sends the file and directory names to Pull Monitor.

4.8.6.9.3 File Transfer Protocol Notification Architecture

Figure 4.8.6.9.3-1 is the File Transfer Protocol Notification architecture diagram. The diagram shows the events sent to the CsFtFTPNotify class and the events the CsFtFTPNotify class sends to other processes. The Class method SetFtpDoneFiles reads the SYSLOG file and extracts the file and directory names involved in the completed transfer.

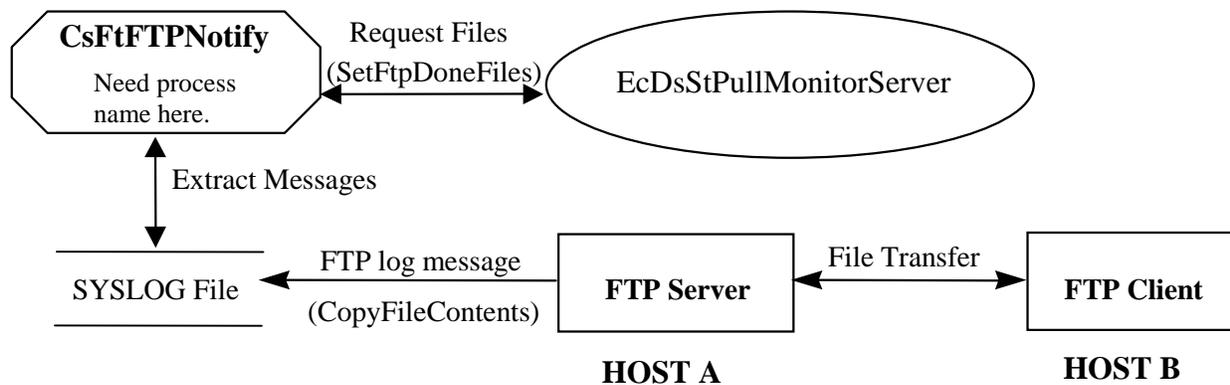


Figure 4.8.6.9.3-1. File Transfer Protocol Notification Architecture Diagram

4.8.6.9.4 File Transfer Protocol Notification Process Descriptions

Table 4.8.6.9.4-1 provides descriptions of the processes shown in the File Transfer Protocol Notification architecture diagram. The CsFtFTPNotify class extracts the file name and location of the files transferred from the SYSLOG file. The information is written to a file supplied by the caller of the class.

Table 4.8.6.9.4-1. File Transfer Protocol Notification Processes

Process	Type	COTS/ Developed	Functionality
FTP Server	Server FTP	COTS	The FTP server running on HOST logs the FTP messages to the SYSLOG file. A copy of the logs is made by the CsFtFTPNotify class
FTP Client	Client FTP	COTS	Initiates the FTP commands and gets or puts the specified file from or to a remote host and copies it to the local host initiating the FTP.

4.8.6.9.5 File Transfer Protocol Notification Process Interface Descriptions

Table 4.8.6.9.5-1 provides descriptions of the interface events shown in the File Transfer Protocol Notification architecture diagram.

Table 4.8.6.9.5-1. File Transfer Protocol Notification Process Interface Events

Event	Event Frequency	Interface	Initiated by	Event Description
File Transfer	One per file transfer	<i>Process:</i> FTP Server	<i>Process:</i> FTP Client Application	The specified file transfer takes place between the client and server.
FTP log message	One per system log file entry	SYSLOG File	<i>Process:</i> FTP Server	The FTP server logs all FTP events to the system log (syslog) file. The CopyFileContents function is used to copy log files.
Extract messages	One per log file copy	SYSLOG File	<i>Process:</i> ? <i>Library:</i> CsFtFTPNotify <i>Class:</i> CsFtFTPNotify	The CsFtFTPNotify class is used to extract messages from the system log (syslog) file.
Request Files	One per FTP request	<i>Function:</i> SetFtpDoneFiles <i>Library:</i> CsFtFTPNotify <i>Class:</i> CsFtFTPNotify	<i>Process:</i> EcDsStPullMonitor Server <i>Class:</i> DsStPullFtpNotify	The EcDsStPullMonitorServer (Pull Monitor) requests a transfer of files via the FTP service by the SetFtpDoneFiles function. After the file transfer, FTPNotify extracts information from the syslog file and gives the file and directory name to the Pull Monitor.

4.8.6.9.6 File Transfer Protocol Notification Data Stores

Table 4.8.6.9.6-1 provides descriptions of the information in the SYSLOG File data store. More detail on these data stores can be found in the Subscription Server Database Design and Schema Specifications for the ECS Project (Refer to CDRL 311).

Table 4.8.6.9.6-1. File Transfer Protocol Data Stores

Data Store	Type	Functionality
SYSLOG File	File	Storage for details of successful or failed file transfers.

4.8.6.10 Remote File Access Group - Bulk Data Server Description

4.8.6.10.1 Bulk Data Server Functional Overview

The Bulk Data Server (BDS) is a non-standard extension to Network File System (NFS), implemented as an enhancement on the client system and a daemon process on the server for transferring large (100 Megabytes and larger) files over high-speed networks. Figure 4.8.6.10.1-1 is a comparison with the NFS/Transmission Control Protocol/Internet Protocol (TCP/IP) protocols in the International Standards Organization (ISO) Open Systems Interconnect (OSI) 7-layer model. BDS exploits the data access speed of the Extended File System (XFS) and data transfer rates of network media, such as HIPPI and fiberchannel, to accelerate standard NFS performance. The BDS protocol, XBDS, modifies NFS functions and reduces the time needed to

transfer large files over a network connection. BDSpro is the Silicon Graphics implementation of XBDS. BDSpro is run on SGI machines with IRIX 6.2 (or later versions) and connected to a high-speed network (such as HIPPI or fiberchannel) running the TCP/IP suite.

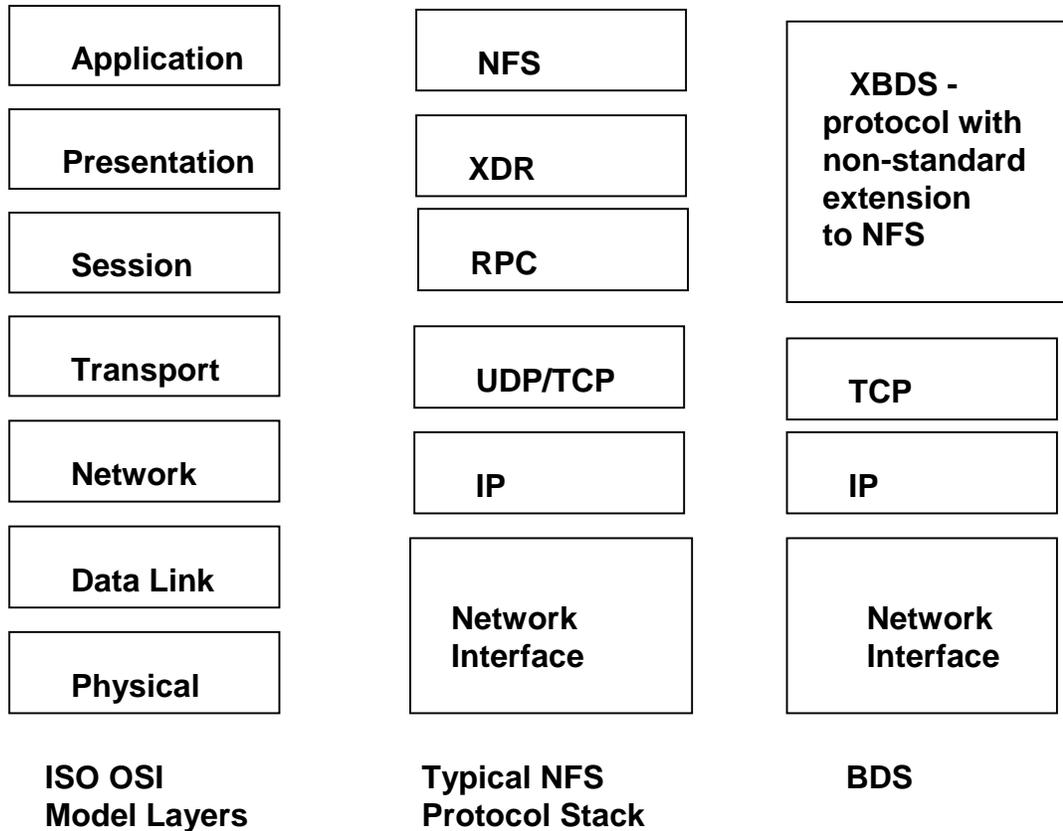


Figure 4.8.6.10.1-1. Bulk Data Server Protocol compared with ONC Protocols

4.8.6.10.2 Bulk Data Server Context

BDS is a file transfer utility to move large data files over the HIPPI communications lines. Figure 4.8.6.10.2-1 is the Bulk Data Server context diagram shown with an ECS application. The BDS applications within the ECS are in the DsStArchiveReal module of the Data Server Storage Management Archive software. The storage location calculation takes the vector of the data file as parameters with the location of the file, the unique file name, the original file name, the size of the file, and a checksum. BDS transfers data files produced by PDPS to archive. The data files are transferred via BDS over HIPPI and stored on AMASS.

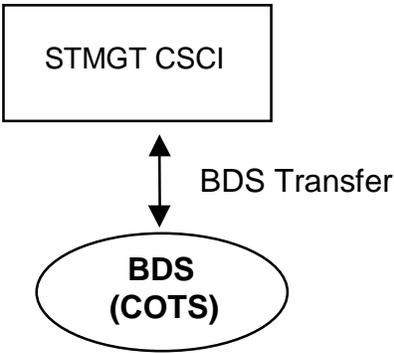


Figure 4.8.6.10.2-1. Bulk Data Server Context Diagram

Table 4.8.6.10.2-1 provides a description of the interface event shown in the Bulk Data Server context diagram.

Table 4.8.6.10.2-1. Bulk Data Server Interface Events

Event	Interface Event Description
BDS transfer	The STMGT CSCI uses the BDS to transfer large data files over the HIPPI interface for storage in the Science Data Server archives.

4.8.6.10.3 Bulk Data Server Architecture

BDS is implemented as an enhancement to the NFS on the client system and as a daemon process on the server. Figure 4.8.6.10.3-1 is the BDS architecture diagram shown over HIPPI on an SGI client-server model.

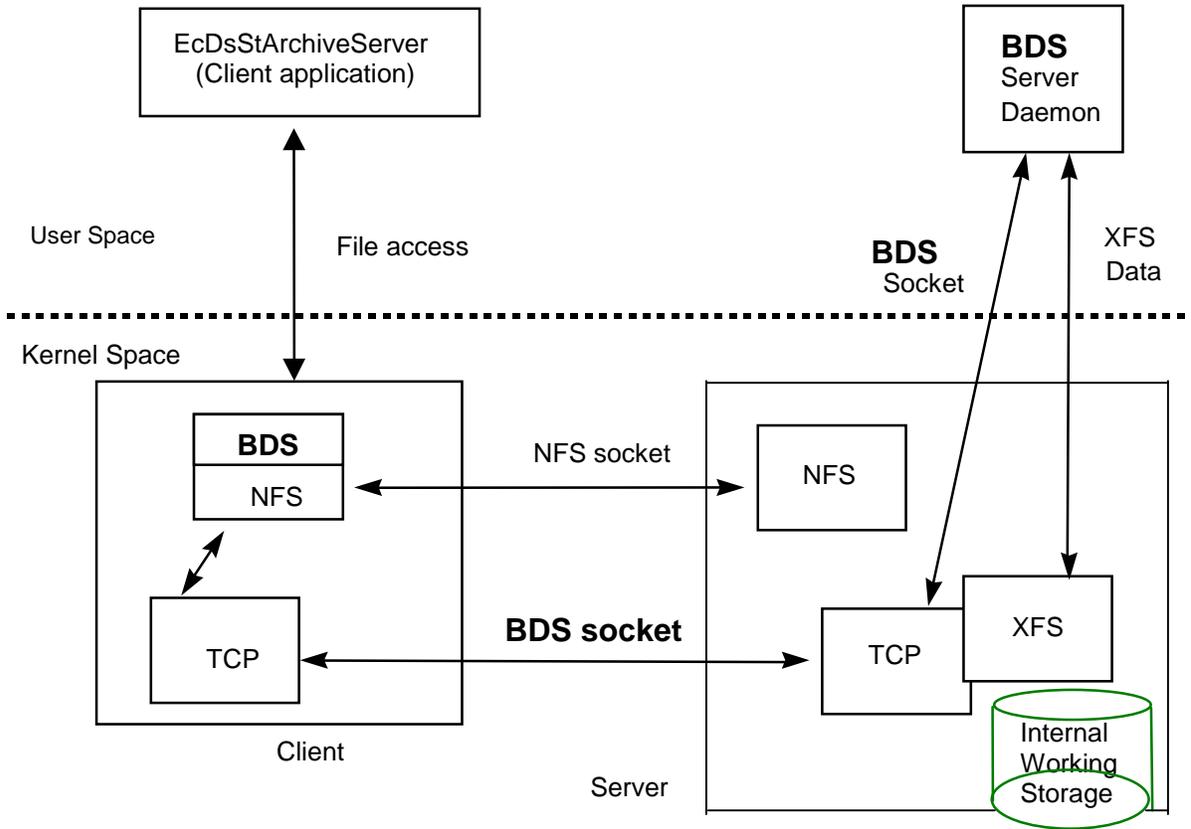


Figure 4.8.6.10.3-1. Bulk Data Server Architecture Diagram

4.8.6.10.4 Bulk Data Server Process Descriptions

Table 4.8.6.10.4-1 provides descriptions of the processes shown in the Bulk Data Server architecture diagram.

Table 4.8.6.10.4-1. Bulk Data Server Processes

Process	Type	COTS/ Developed	Functionality
Client application	Client	COTS	Provides XBDS protocol for client functions implemented as enhanced NFS/ External Data Representation (XDR)/RPC protocols.
BDS server daemon	Server	COTS	Provides XBDS protocol server functions, implemented as enhanced protocols for NFS/XDR/RPC protocols.

4.8.6.10.5 Bulk Data Server Process Interface Descriptions

Table 4.8.6.10.5-1 provides descriptions of the interface event shown in the Bulk Data Server architecture diagram.

Table 4.8.6.10.5-1. Bulk Data Server Process Interface Events

Event	Event Frequency	Interface	Initiated by	Event Description
File access	One per file transfer	BDS (COTS)	Process: EcDsStArchiveServer	The EcDsStArchiveServer uses the BDS to transfer large data files over the HIPPI interface for storage in the Science Data Server archives.

4.8.6.10.6 Bulk Data Server Data Stores

Data stores are not applicable for the Bulk Data Server.

4.8.6.11 Remote File Access Group - Network File System Description

4.8.6.11.1 Network File System Functional Overview

The Network File System (NFS) provides a file sharing system between computers. NFS consists of a mounting protocol with a server, a file locking protocol with a server, and daemons to coordinate the file services provided. A server exports (or shares) a system of files by providing file system access to other hosts on a common network. An NFS client must explicitly mount the file system of interest before the file system is made accessible.

4.8.6.11.2 Network File System Context

Figure 4.8.6.11.2-1 is the Network File System context diagram. The NFS mounted directories reside on mount points made accessible for the use of other hosts machines. Table 4.8.6.11.2-1 provides descriptions of the interface event shown in the context diagram.

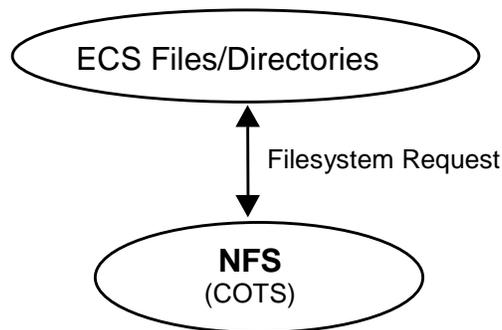


Figure 4.8.6.11.2-1. Network File System Context Diagram

Table 4.8.6.11.2-1. Network File System Interface Events

Event	Interface Event Description
Filesystem Request	The NFS clients request files or directories via an established mount point. The NFS Server makes the storage device(s) and its data accessible for use by the clients.

4.8.6.11.3 Network File System Architecture

Figure 4.8.6.11.3-1 is the Network File System architecture diagram. The diagram shows the file requests are via system calls from the Virtual File Server (VFS). The NFS client uses the XDR/RPC and networking.

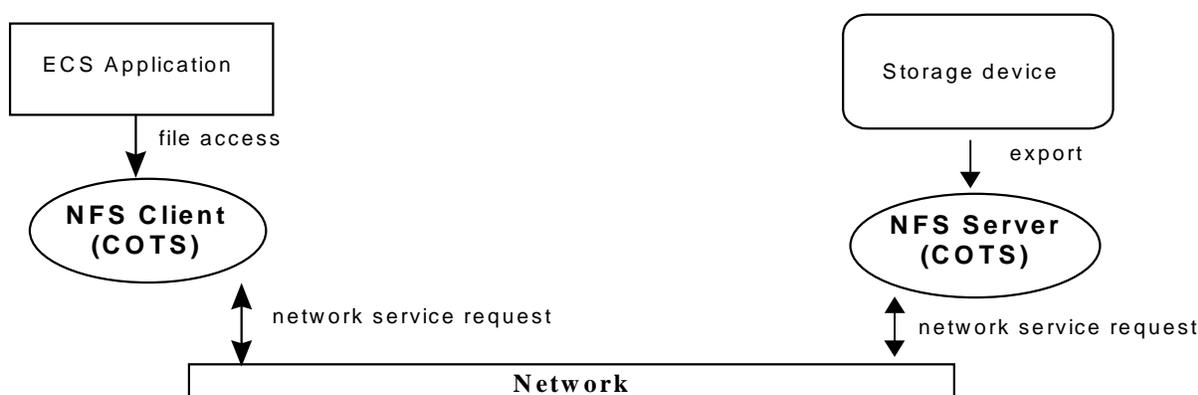


Figure 4.8.6.11.3-1. Network File System Architecture Diagram

4.8.6.11.4 Network File System Process Descriptions

Table 4.8.6.11.4-1 provides descriptions of the processes shown in the Network File System architecture diagram.

Table 4.8.6.11.4-1. Network File System Processes

Process	Type	COTS/ Developed	Functionality
NFS client	Client	COTS	The Target host providing the mounts.
NFS server	Server	COTS	The Source host exporting the data.

4.8.6.11.5 Network File System Process Interface Descriptions

Table 4.8.6.11.5-1 provides the descriptions of the interface events shown in the NFS architecture diagram.

Table 4.8.6.11.5-1. Network File System Process Interface Events

Event	Event Frequency	Interface	Initiated by	Event Description
File Access	One per file access	<i>NFS Client (COTS)</i>	<i>ECS applications</i>	The application uses the file accessed via the NFS client.
Export	One per server export	<i>NFS Server (COTS)</i>	<i>Storage Device</i>	The NFS server exports the details of storage devices to verify clients.

4.8.6.11.6 Network File System Data Stores

Data stores are not applicable for the Network File System.

4.8.6.12 Remote File Access Group - Filecopy Description

4.8.6.12.1 Filecopy Functional Overview

Filecopy is the utility to copy large files from a specified source location to a specified destination location with the option of compression and decompression. The utility uses the gzip option to reduce the file size using the Lampel-Ziv coding (LZ77) technique. For Decompression, it uses the gunzip option to return the file to its original size. The EcUtCopyExec utility uses Unix read/write commands to actually copy the large file and in the event of NFS time errors, the utility retries ten times with a five-second-time delay in between retries.

4.8.6.12.2 Filecopy Context

The Filecopy utility is used by the STMGT CSCI to copy files from the INGEST staging disk to the archive and from the archives to the Read Only Cache (RAID Disk Array). Also, the MCF Files are copied from the SDSRV to the DDIST staging Disk using Filecopy. Figure 4.8.6.12.2-1 is the Filecopy context diagram.

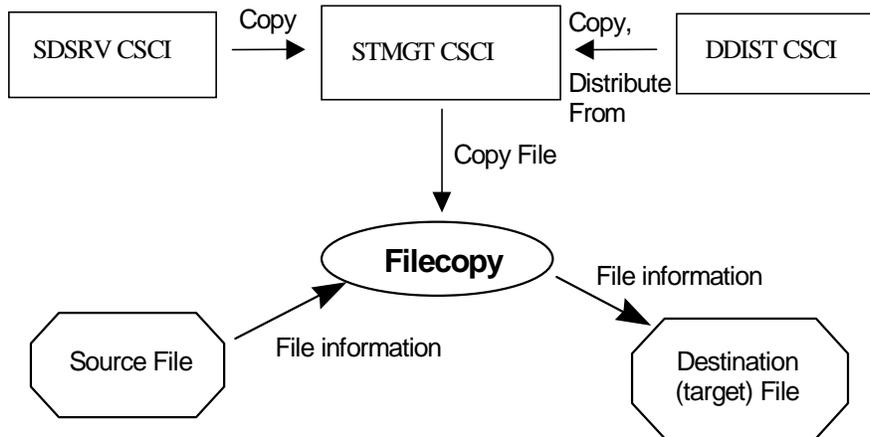


Figure 4.8.6.12.2-1. Filecopy Context Diagram

Table 4.8.6.12.2-1 provides descriptions of the interface events shown in the Filecopy context diagram.

Table 4.8.6.12.2-1. Filecopy Interface Events

Event	Interface Event Description
Copy	The DDIST CSCI sends requests to the STMGT CSCI to copy data within staging disks and between staging disks.
Distribute From	The DDIST CSCI sends requests to the STMGT CSCI to copy files from staging disks to an allocated peripheral resource.
Copy File	The STMGT CSCI inserts data into the archives sending a request for a Unix file copy into the AMASS cache by buffered read/write software using the Filecopy utility.
File Information	File information includes source location, destination location, and file size.

4.8.6.12.3 Filecopy Architecture

The Filecopy utility uses options to provide copy features of file compression, decompression, or the standard copy. Figure 4.8.6.12.3-1 is the Filecopy architecture diagram. The diagram shows the events sent to the FileCopy class and the events the FileCopy class sends to update the directories.

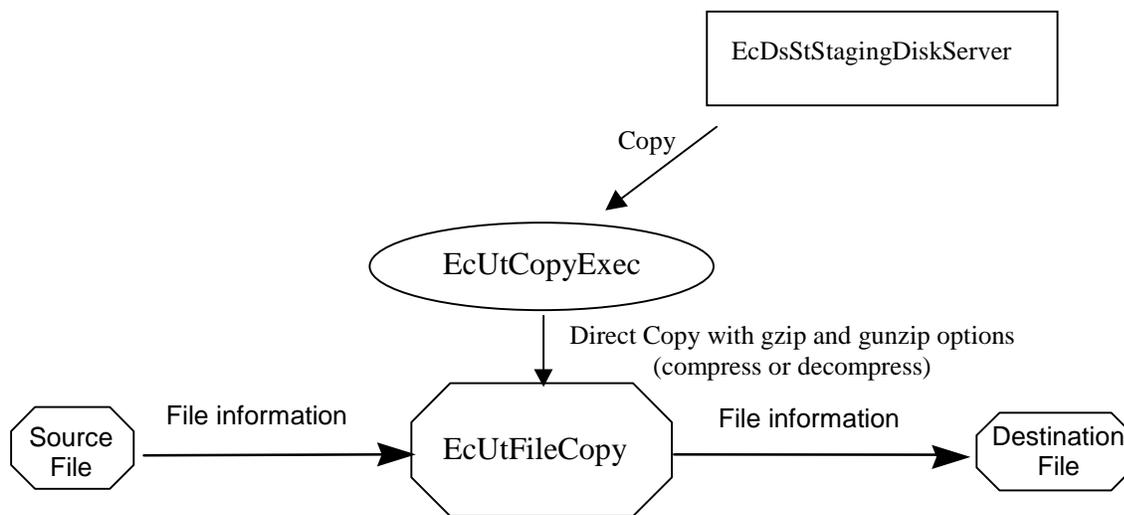


Figure 4.8.6.12.3-1. Filecopy Architecture Diagram

4.8.6.12.4 Filecopy Process Descriptions

Table 4.8.6.12.4-1 provides descriptions of the processes shown in the Filecopy architecture diagram. The EcUtFileCopy utility class copies files, with copy options, from one specified location to another.

Table 4.8.6.12.4-1. Filecopy Process

Process	Type	COTS/ Developed	Functionality
EcUtCopyExec	Utility	Developed	Used for direct copy of files with timeout checks and retry features.

4.8.6.12.5 Filecopy Process Interface Descriptions

Table 4.8.6.12.5-1 provides descriptions of the interface events shown in the FileCopy architecture diagram.

Table 4.8.6.12.5-1. Filecopy Process Interface Events

Event	Event Frequency	Interface	Initiated by	Event Description
Copy	One file copy per request	<i>Process:</i> EcUtCopyExec <i>Classes:</i> DsStArchiveReal, DsStStagingDisk, DsStStagingMonitorReal, DpPrDataManager	<i>Process:</i> EcDsStStagingDiskServer <i>Library:</i> DsStSt <i>Class:</i> DsStStagingDisk	The EcDsStStagingDiskServer copies data within staging disks and between staging disks using the EcUtCopyExec process.
Direct Copy (with gzip/gnuzip options)	One per direct copy	<i>binary:</i> EcUtFilecopy main function	<i>Process:</i> EcUtCopyExec <i>Library:</i> EcUtMisc <i>Class:</i> EcUtFilecopy	EcUtCopyExec is used for making direct copy. Checks for NFS timeout and retries when an error is encountered. Also checks for compression type specified with the compress/decompress option.
File Information	One per file copy	Source File, Destination File	<i>Process:</i> EcUtCopyExec <i>Library:</i> EcUtMisc <i>Class:</i> EcUtFilecopy	File information contains data file location information.

4.8.6.12.6 Filecopy Data Stores

Data stores are not applicable for the Filecopy service.

4.8.6.13 Mail Support Group Description

4.8.6.13.1 E-mail Server Functional Overview

The E-mail server provides an interactive and a development interface for managing the electronic mail functions. The interactive interface is implemented with COTS products and provides send, receive, and read message functionality. The development interfaces, or Application Programming Interfaces (APIs), are limited to sending messages.

4.8.6.13.2 E-mail Server Context

Figure 4.8.6.13.2-1 is the E-mail Server context diagram. Table 4.8.6.13.2-1 provides descriptions of the interface events shown in the E-mail Server context diagram.

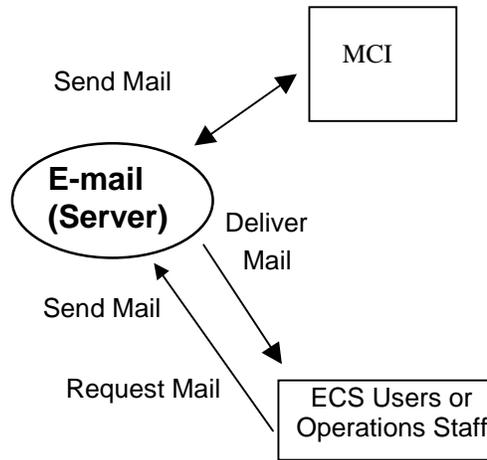


Figure 4.8.6.13.2-1. E-mail Server Context Diagram

Table 4.8.6.13.2-1. E-mail Server Interface Events

Event	Interface Event Description
Send Mail	The MCI uses the development interface to send mail and the API spawns a <i>sendmail</i> process to deliver the message. Interactive users, use the COTS software product, which delivers the mail message.
Deliver Mail	The mail server delivers the mail to the addressed user.
Request Mail	ECS users or operations staff sends requests for e-mail messages to the E-mail Server.

4.8.6.13.3 E-mail Server Architecture

The E-mail server is a COTS software product. Figure 4.8.6.13.3-1 is the E-mail server architecture diagram.

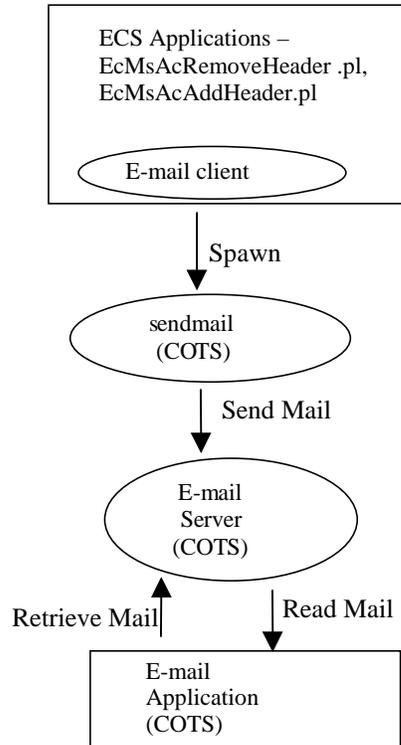


Figure 4.8.6.13.3-1. E-mail Server Architecture Diagram

4.8.6.13.4 E-mail Server Process Descriptions

Table 4.8.6.13.4-1 provides descriptions of the processes shown in the E-mail Server architecture diagram.

Table 4.8.6.13.4-1. E-mail Server Processes

Process	Type	COTS/ Developed	Functionality
E-mail Client	Other	Developed	The E-mail client is a library used by ECS applications to send electronic mail. The E-mail client provides APIs for creating E-mail messages and spawns a <i>sendmail</i> process to deliver the mail to the mail server.
Sendmail	Other	COTS	Sendmail is a COTS software product spawned by the E-mail client when E-mail is ready to send. The SMTP protocol is used to send the E-mail to the E-mail server.
E-mail Server	Server	COTS	A COTS software E-mail server product.
E-mail Application	Other	COTS	A COTS software product for sending, receiving, and reading E-mail.

4.8.6.13.5 E-mail Server Process Interface Descriptions

Table 4.8.6.13.5-1 provides descriptions of the interface events shown in the E-mail server architecture diagram.

Table 4.8.6.13.5-1. E-mail Server Process Interface Events

Event	Event Frequency	Interface	Initiated by	Event Description
Spawn	One per process spawned	<i>Library:</i> libc API: System ()	<i>Processes:</i> EcMsAcRemoveHeader.pl, EcMsAcAddHeader.pl <i>Library:</i> CsEmMailRelA <i>Class:</i> CsEmMailRelA	The EcMsAcRemoveHeader.pl and EcMsAcAddHeader.pl scripts invoke a process to send E-mail via an API.
Send mail	One per E-mail send	<i>Process:</i> E-mail Server (COTS)	<i>Process:</i> Sendmail (COTS)	A command from the sendmail process to send electronic mail routed via the E-mail server.
Read mail	One per e-mail read	<i>Process:</i> E-mail Application (COTS)	<i>Process:</i> E-mail Server (COTS)	E-mail is received from the COTS application (sendmail) and routed to another user via the E-mail server.
Retrieve mail	One per E-mail send	<i>Process:</i> E-mail Server (COTS)	<i>Process:</i> E-mail Application (COTS)	E-mail is received from an ECS application, a sendmail process is spawned to forward the mail, and the E-mail is sent via the E-mail server to another user.

4.8.6.13.6 E-mail Server Data Stores

Data stores are not applicable for the E-mail Server.

4.8.6.14 Virtual Terminal Description

4.8.6.14.1 Virtual Terminal Functional Overview

The Virtual Terminal (VT) effectively hides the terminal characteristics and data handling conventions from both the server host and Operations staff, and enables the Operations staff to remotely log on to other ECS machines. The CSS provides the kerberized telnet and the telnetd on available systems and common capability support for the ECS dial-up service.

4.8.6.14.2 Virtual Terminal Context

The CSS provides the kerberized telnet and the telnetd to the ECS systems. Telnet and telnetd (non-kerberized) are distributed as part of the operating system provided. The dial-up service provides users with access to the ECS character-based user interface (CHUI) search and order

tool. Figure 4.8.6.14.2-1 is the Virtual Terminal context diagram and Table 4.8.6.14.2-1 provides the descriptions of the interface events shown in the Virtual Terminal context diagram.

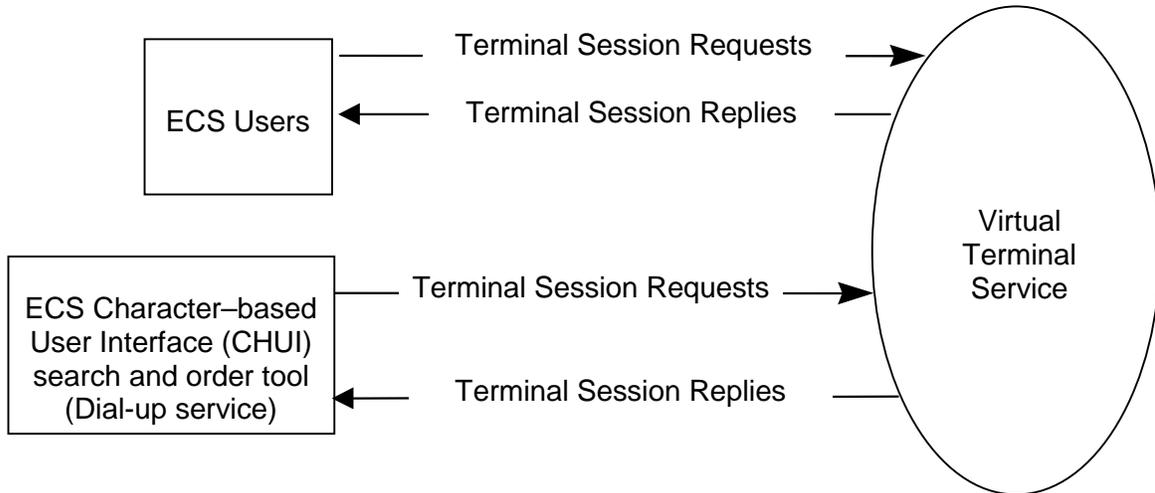


Figure 4.8.6.14.2-1. Virtual Terminal Context Diagram

Table 4.8.6.14.2-1. Virtual Terminal Interface Events

Event	Interface Event Description
Terminal Session Requests (Dial-up Service)	ECS users request a connection to a specified host via dial-up service.
Terminal Session Requests (ECS Users)	ECS users request a telnet session with a specified ECS host.
Terminal Session Replies (from the VT Service to ECS or other Dial-Up Users)	The VT Server residing on the ECS host responds to the terminal session requests and interacts via the successful connection.

4.8.6.14.3 Virtual Terminal Architecture

Figure 4.8.6.14-3-1 is the Virtual Terminal architecture diagram. The diagram shows the event traffic between the Telnet with ECS Users and Telnet with Dial-up users.

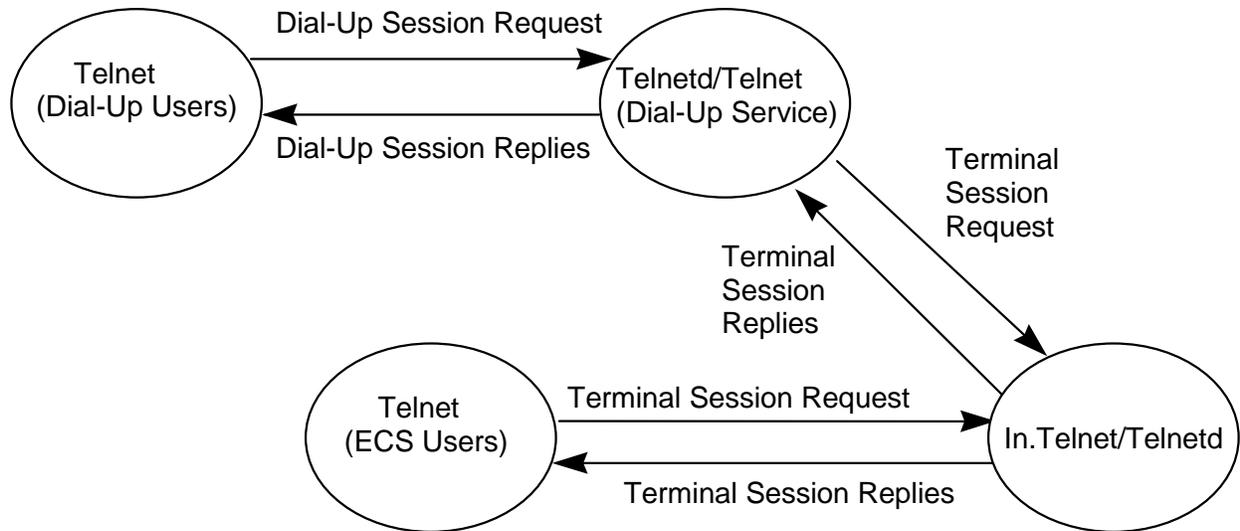


Figure 4.8.6.14.3-1. Virtual Terminal Architecture Diagram

4.8.6.14.4 Virtual Terminal Process Descriptions

Table 4.8.6.14.4-1 provides the descriptions of the processes shown in the Virtual terminal architecture diagram.

Table 4.8.6.14.4-1. Virtual Terminal Processes

Process	Type	COTS/ Developed	Functionality
Telnet (Dial-up Users)	Client	COTS	Provides the dial-up terminal session as requested on the client-side via dial-up service.
Telnet (ECS Users)	Client	COTS	Provides the user interface to a remote system using the TELNET protocol.
Telnetd/Telnet (Dial-up Service)	Server/Client	COTS	Enables users to interact with the host through a dial-up service.
Telnetd or in.telnetd	Server	COTS	Function provides servers supporting TELNET with virtual terminal protocol.

4.8.6.14.5 Virtual Terminal Process Interface Descriptions

Table 4.8.6.14.5-1 provides the descriptions of the interface events shown in the Virtual Terminal architecture diagram.

Table 4.8.6.14.5-1. Virtual Terminal Process Interface Events

Event	Event Frequency	Interface	Initiated by	Event Description
Dial-up Session Request	One per connection request	<i>Process:</i> Telnetd/Telnet (dial-up service)	<i>Process:</i> Telnet (COTS – dial-up users)	Any ECS user requiring a logon to another machine from the current machine. Users request to establish connection to a specified host via dial-up.
Dial-up Session Replies	One per session reply	<i>Process:</i> Telnet (dial-up service)	<i>Process:</i> Telnetd/Telnet (dial-up service)	The Dial-up service provides users a dial-up session to request a terminal session to the host's telnetd.
Terminal Session Request (Telnet kerberized request)	One per request to establish a session	<i>Process:</i> In.Telnet/Telnetd	<i>Process:</i> Telnetd/Telnet (dial-up service)	Either the user or dial-up service requests to establish a telnet session with the specified host.
Terminal Session Replies (Kerberos Telnetds)	One per connection request	<i>Process:</i> Telnet (ECS users)	<i>Process:</i> In.Telnet/Telnetd	The Host Virtual Terminal Process, Kerberos telnetd, responds to the connection requests and establishes or maintains the sessions.

4.8.6.14.6 Virtual Terminal Data Stores

Data stores are not applicable for the Virtual Terminal.

4.8.6.15 Cryptographic Management Interface Software Description

4.8.6.15.1 Cryptographic Management Interface Functional Overview

The Cryptographic Management Interface (CMI) classes provide the requesting process with a server account and a randomly generated password so the server can access non-DCE services (i.e., Sybase). These passwords (and optionally login names) are generated dynamically based on a pseudo-random number used as the seed for the password.

4.8.6.15.2 Cryptographic Management Interface Context

Figure 4.8.6.15.2-1 is the Cryptographic Management Interface context diagram. Servers (PF or non-PF) use the CMI with a need for access to non-DCE services. Table 4.8.6.15.2-1 provides descriptions of the interface events shown in the Cryptographic Management Interface context diagram.

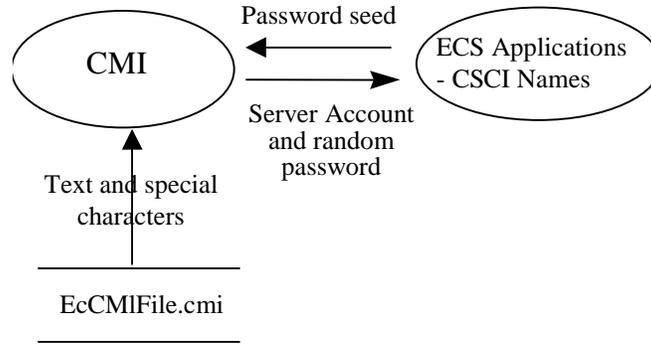


Figure 4.8.6.15.2-1. Cryptographic Management Interface Context Diagram

Table 4.8.6.15.2-1. Cryptographic Management Interface Events

Event	Interface Event Description
Password seed	The ECS application (PLANG CSCI) requests an account and provides a password seed to CMI.
Server account and random password	Account with random passwords created for the server is passed back to the server.
Text and special characters	Text and special characters read from a file for password generation.

4.8.6.15.3 Cryptographic Management Interface Architecture

Figure 4.8.6.15.3-1 is the Cryptographic Management Interface (CMI) architecture diagram. The diagram shows the event traffic between the CMI process and the ECS applications that interact with CMI for database connections.

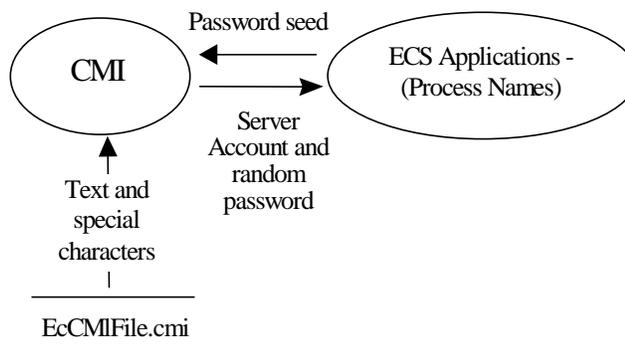


Figure 4.8.6.15.3-1. Cryptographic Management Interface Architecture Diagram

4.8.6.15.4 Cryptographic Management Interface Process Descriptions

Table 4.8.6.15.4-1 provides descriptions of the processes shown in the Cryptographic Management Interface context diagram.

Table 4.8.6.15.4-1. Cryptographic Management Interface Processes

Process	Type	COTS/ Developed	Functionality
ECS Process Names	Server	Developed	Requests account with random password for access to non-DCE services.
CMI	Other	Developed	A server account and randomly generated password are returned to the requesting server.

4.8.6.15.5 Cryptographic Management Interface Process Interface Descriptions

Table 4.8.6.15.5-1 provides the descriptions of the interface events shown in the Cryptographic Management Interface architecture diagram.

Table 4.8.6.15.5-1. Cryptographic Management Interface Process Interface Events

Event	Event Frequency	Interface	Initiated by	Event Description
Password seed	One per password seed	<i>Process:</i> CMI <i>Library:</i> EcSeCmi <i>Class:</i> EcSeCmi	<i>ECS Applications</i> <i>Library:</i> ? <i>Class:</i> ?	The server provides a unique number as a seed for generating a password.
Server Account and random password	One per account and password	ECS Applications <i>Library:</i> ? <i>Class:</i> ?	<i>Process:</i> CMI <i>Library:</i> EcSeCmi <i>Class:</i> EcSeCmi	CMI generates a random password for the account based on the seed.

4.8.6.15.6 Cryptographic Management Interface Data Stores

Table 4.8.6.15.6-1 provides descriptions of the data store shown in the Cryptographic Management Interface architecture diagram.

Table 4.8.6.15.6-1. Cryptographic Management Interface Data Stores

Data Store	Type	Functionality
EcCMIFile.cmi	Other	This is a flat file of textual and special characters used by the CMI password generation algorithm to create passwords.

4.8.6.16 Domain Name Server Software Description

4.8.6.16.1 Domain Name Server Functional Overview

Domain Name Server (DNS) performs name-to-address and address-to-name resolution between hosts within the local administrative domain and across domain boundaries. DNS is COTS software implemented as server by running a daemon called “in.named.” Servers running the in.named daemon are referred to as name servers.

The server is implemented through a resolver instead of a daemon from the client side. The function of in.named is to resolve user queries for device names or addresses (DNS requires the address of at least one name server to be in the file /etc/resolv.conf). The name server, when queried for a name or an address, returns the answer to the query or a referral to another name server to query for the answers.

Each domain uses at least two kinds of DNS servers (primary and secondary) to maintain the name and address data corresponding to the domain. The primary server keeps the master copy of the data when it starts up in the “in.named,” daemon and delegates authority to other servers both inside and outside of its domain. A secondary server maintains a copy of the name and address data for the domain. When secondary server boots in.named, it requests the data for a given domain from the primary server. The secondary server then checks with the primary server periodically and requests updates to the daemon data so the secondary server is kept up to date with the primary.

DNS namespace is hierarchically organized, with nested domains, like directories. The DNS namespace consists of a tree of domains. Figure 4.8.6.16.1-1 is an Internet domain hierarchy diagram. The top-level domains are edu, arpa, com, gov, net, and for simplicity, not showing org, mil, and int, at the root level. The second level domain is nasa for gov. The third level domain is ecs for the ECS project for nasa.gov.

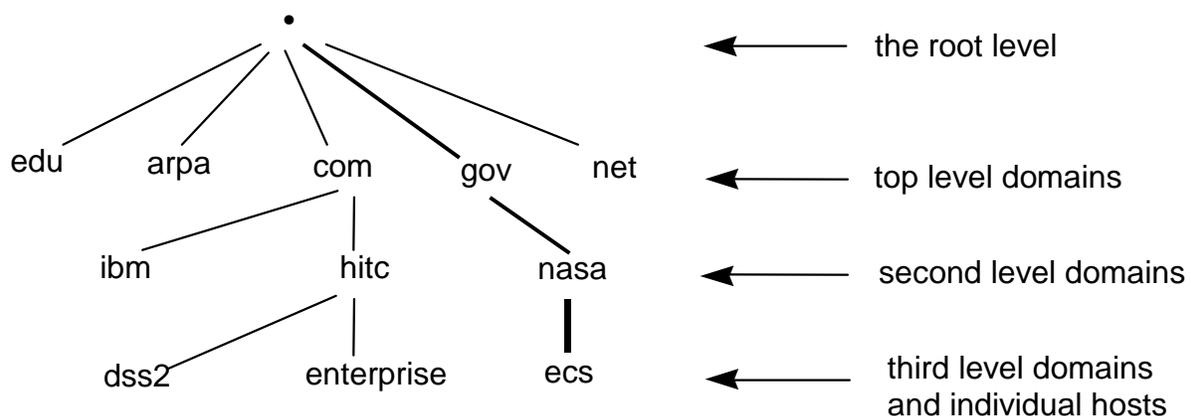


Figure 4.8.6.16.1-1. Domains Hierarchy Diagram

The fourth level domains in the ECS project include domains of DAACs: gsfcb, gsfcmo, and etc. Figure 4.8.6.16.1-2 is the hierarchy diagram of the fourth level domains in the ECS project. The

DAAC and M&O domains are part of the overall DNS. The top-level domain is `ecs.nasa.gov` and the two lower level domains for the DAACs, for example, `gsfcb.ecs.nasa.gov` and `gsfcmo.ecs.nasa.gov` for the GSFC DAAC. The former is for the Version 2. 0-production network and the latter are for the GSFC M&O network.

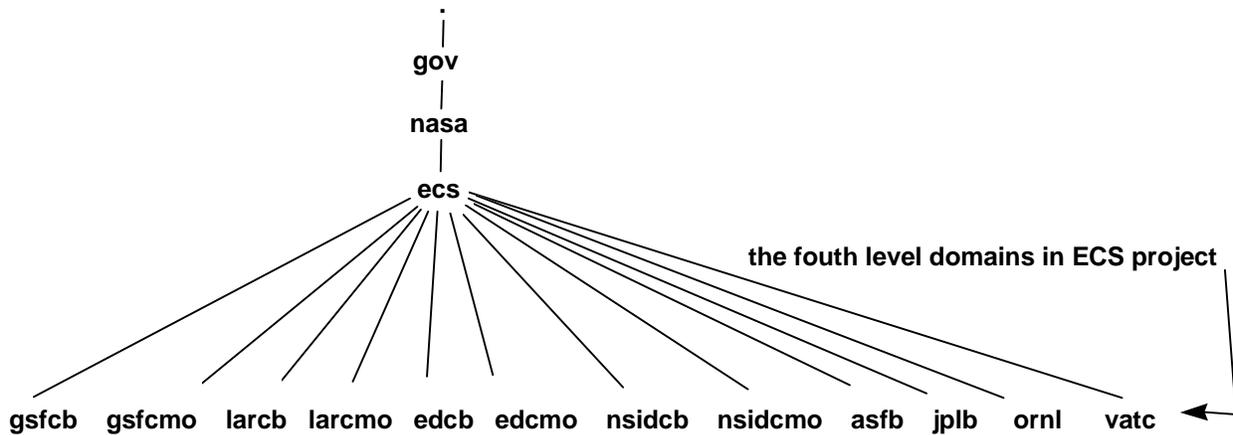


Figure 4.8.6.16.1-2. DNS Domains of the ECS Project Diagram

Figure 4.8.6.15.1-3 is the ECS topology domain diagram.

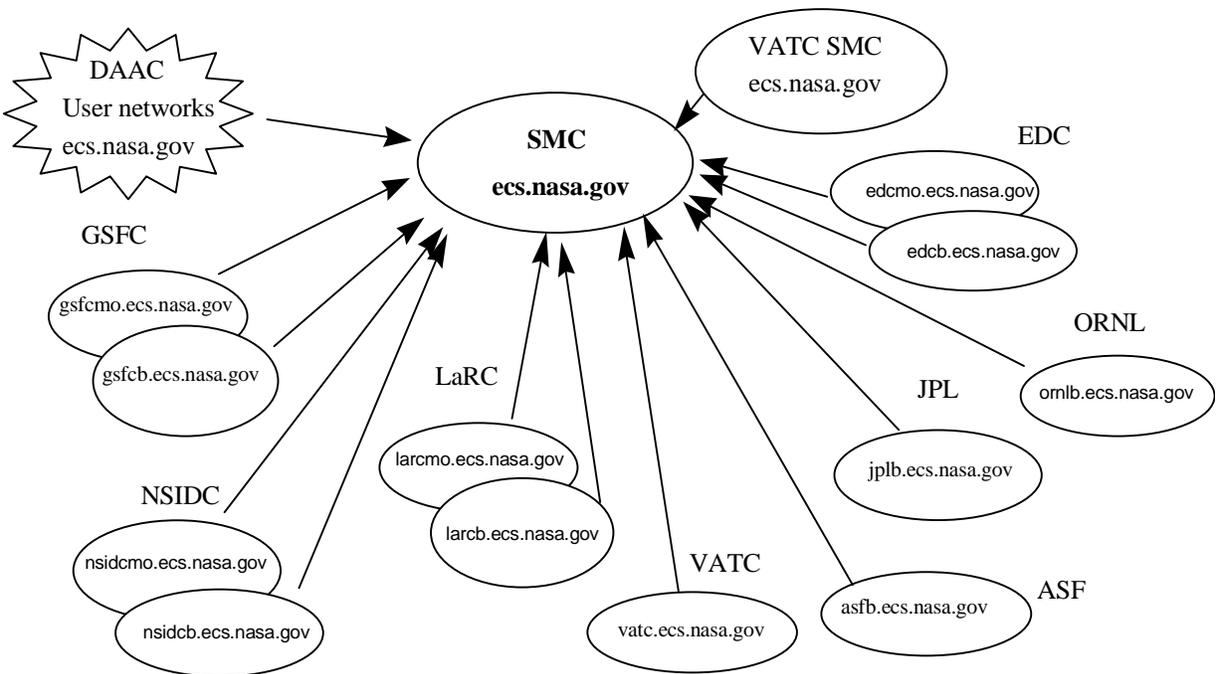


Figure 4.8.6.16.1-3. ECS Topology Domains Diagram

4.8.6.16.2 Domain Name Server Context

Figure 4.8.6.16.2-1 is the Domain Name Server context diagram.

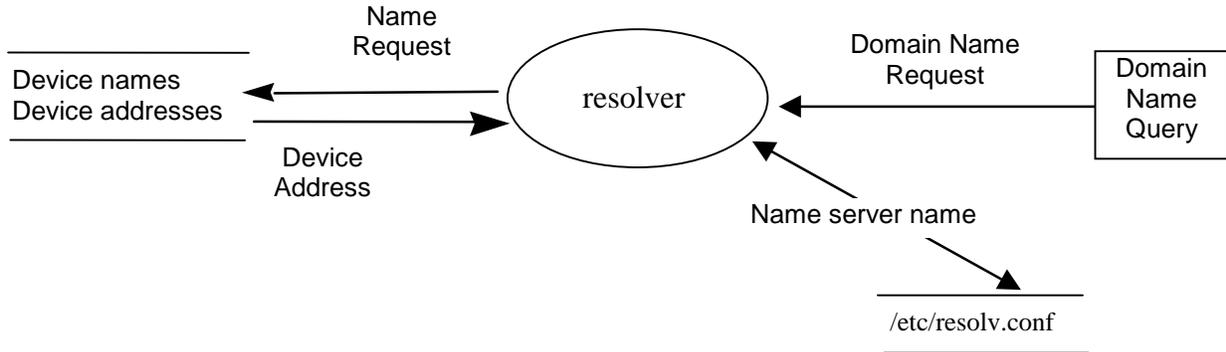


Figure 4.8.6.16.2-1. Domain Name Server Context Diagram

4.8.6.16.3 Domain Name Server Architecture

The Domain Name Server architecture diagram is the same as the context diagram and is not duplicated here. When the DNS client has a request for data, it searches the servers listed in the `/etc/resolv.conf` file in the order the servers were added to the file. When the first server does not contain the information of interest for the client, the second server in the list is searched and the search continues until the information is found.

4.8.6.16.4 Domain Name Server Process Descriptions

Table 4.8.6.16.4-1 provides descriptions of the Domain Name Server processes shown in the Domain Name Server context diagram.

Table 4.8.6.16.4-1. Domain Name Server Process

Process	Type	COTS/ Developed	Functionality
resolver	Client	COTS	Searches data store of device names and device addresses for information requested in the Domain Name Request. First entry in the file <code>/etc/resolv.conf</code> is used as the place to start searching.

4.8.6.16.5 Domain Name Server Process Interface Descriptions

Table 4.8.6.16.5-1 provides descriptions of the interface events shown in the Domain Name Server architecture diagram.

Table 4.8.6.16.5-1. Domain Name Server Process Interface Events

Event	Event Frequency	Interface	Initiated by	Event Description
Domain Name Request	One per user request	<i>COTS Software:</i> resolver	User	A DNS user requests data.
Name server name	One per search directory change	Data Store	<i>COTS Software:</i> resolver	The resolver retrieves the pathname for the directory to search for the user requested data from the /etc/resolv.conf database table. New file names are added to the list in the order they are stored.
Device Address	One per resolved address	<i>COTS Software:</i> resolver	<i>COTS Software:</i> name server	Returns the resolved address to the domain name requester via the Resolver.
Name Request	One per domain name request	<i>COTS Software:</i> name server	<i>COTS Software:</i> resolver	The resolver retrieves the domain name (device name and address) for the name server from an internal file used by the COTS software.

4.8.6.16.6 Domain Name Server Data Stores

Table 4.8.6.16.6-1 provides descriptions of the data store shown in the Domain Name Server architecture diagram.

Table 4.8.16.6-1. Domain Name Server Data Stores

Data Store	Type	Functionality
/etc/resolv.conf	Other	Stores the primary and secondary server names.

4.8.6.17 Infrastructure Libraries Group Description

4.8.6.17.1 Infrastructure Libraries Group Functional Overview

The Infrastructure Library Group (ILG) is a library of reusable software frameworks and infrastructures used by ECS servers configured as a distributed client-server system. Table 4.8.6.17.1-1 provides descriptions of the infrastructures in the ILG.

Table 4.8.6.17.1-1. Infrastructure Libraries (1 of 2)

Library	Description
Process Framework (PF)	Provides an extensible mechanism for transparent incorporation of features, from a library of infrastructure features, such as mode management, error and event logging, life-cycle services, and OODCE directory and security services into ECS application processes.
Server Request Framework (SRF)	Provides enhancements to OODCE RPC functionality by providing synchronous and asynchronous message passing capabilities and with persistent storage on request.

Table 4.8.6.17.1-1. Infrastructure Libraries (2 of 2)

Library	Description
Universal References (UR)	Provides a mechanism, usable system wide, to uniquely identify the address space of ECS data and service objects (locally and remote).
Error/Event Logging	Provides the mechanism for logging application errors and system events for MSS application monitoring.
Message Passing (MP)	Provides peer-to-peer synchronous and asynchronous communications with store, forward, and persistence features.
ServerUR	Provides unique (Universal Reference) identification for data and service objects in ECS.
Fault Handling (FH)	Provides fault recovery capabilities by enabling clients to reconnect with a server after a prior connection is lost.
DBWrapper directory	The DBWrapper directory is the DBMS Interface Infrastructure Library used by ECS applications to connect to the Sybase Servers. Sybase SQL servers operating under the DBMS, operate by ECS defined guidelines for mode management, thread safety, error handling, error recovery, security, configuration management, and performance of database connections.

4.8.6.17.2 Infrastructure Libraries Group Context

A context diagram is not applicable to the Infrastructure Libraries Group.

4.8.6.17.3 Infrastructure Libraries Group Architecture

An architecture diagram is not applicable to the Infrastructure Libraries Group.

4.8.6.17.4 Infrastructure Libraries Group Process Descriptions

Descriptions of the individual processes in the Infrastructure Libraries Group are not applicable.

4.8.6.17.5 Infrastructure Libraries Group Interface Descriptions

Table 4.8.6.17.5-1 provides descriptions of the interfaces the Infrastructure Libraries Group.

Table 4.8.6.17.5-1. Infrastructure Libraries Group Interfaces (1 of 2)

Library	Interface	Initiated by	Library Description
Process Framework (PF)	<p><i>Library:</i> EcPf</p> <p><i>Classes:</i> EcPfManagedServer, EcPfclient</p>	<p>EcDsScienceDataServer, EcDsHdfEosServer, EcDsDistributionServer, EcDsStArchiveServer, EcDsStStagingMonitorServer, EcDsStStagingDiskServer, EcDsSt8MMServer, EcDsStD3Server, EcDsStIngestFtpServer, EcDsStFtpDisServer, EcDsStPrintServer, EcDsStPullMonitorServer, EcInAuto, EcInPolling, EcInReqMgr, EcInMailGWServer, EcInGran, EcCIWbJdt, EcCIDtUserProfileGateway, EcDmDictServer, EcDmDdMaintenanceTool, EcDmLimServer, EcDmV0ToEcsGateway, EcDmEcsToV0Gateway, EcDmAsterToEcsGateway, EcDmEcsToAsterGateway, EcIoAdServer, EcPISubMgr, EcPIOdMgr, EcDpPrDeletion, EcDpPrJobMgmt, EcDpPrJobMgmtClient, EcSbSubServer, EcGwDARServer, EcCsEmailParser, EcCsLandsat7Gateway, EcCsMojoGateway, EcMsAcRegUserSrvr, EcMsAcOrderSrvr, EcCIWbFoliodProxyServer, EcCIWbJessProxyServer</p>	<p>Provides flexible mechanism (encapsulation) for ECS client and server applications to transparently include ECS infrastructure features from a library of services. The PF process is the encapsulation of an object with ECS infrastructure features and therefore the encapsulated object is fully equipped with the attributes needed to perform the activities assigned. Features and services include:</p> <ul style="list-style-type: none"> • Mode management • Error and event logging • Life-cycle services • OODCE directory and security services

Table 4.8.6.17.5-1. Infrastructure Libraries Group Interfaces (2 of 2)

Library	Interface	Initiated by	Library Description
Server Request Framework (SRF)	<i>Library(Common):</i> Srf <i>Classes:</i> EcSrRequestServer_C, EcSrAsynchRequest_C	Any application intending to use asynchronous messaging capabilities	Provides enhancements to OODCE RPC functionality by providing a synchronous and asynchronous message passing capability. Message requests can be persistently stored.
Universal References (UR)	<i>Library(Common):</i> EcUr	Object Origination	Provides a system wide unique identification for ECS data and service objects.
Error/Event Logging	<i>Library:</i> Event <i>Class:</i> EcLgErrorMsg	Any application requiring logging	Provides a mechanism for logging application errors and system events for MSS application monitoring.
Message Passing (MP)	<i>Library:</i> EcDcMsgPsgn1	Any application intending to use asynchronous/ synchronous messaging capabilities	Provides a peer-to-peer synchronous and asynchronous communications with store, forward, and persistence storage.
ServerUR	<i>Library(Common):</i> EcUr <i>Class:</i> EcUrServerUR	<i>Processes:</i> EcDmDdMaintenanceTool EcSbGui <i>Classes:</i> EcNsServiceLoc, EcCsMojoGateway, EcCISubscriptionclient, EcSbSubscriptionRserver, DSS <i>Libraries:</i> DsBt, DsDe1, DsGe	Provides a mechanism for a server resource to be uniquely identified in the ECS.
Fault Handling (FH)	<i>Library:</i> EcFh <i>Class:</i> EcFhExecutor	All ECS applications that instantiate client proxies and make RPCs to servers	Provides fault recovery capabilities by enabling clients to reconnect with servers after losing a prior connection.
DBWrapper directory	<i>Libraries:</i> EcPoDbRW, EcPoDb <i>Class:</i> EcPoConnectionsRW	<i>Processes:</i> EcDsStArchiveServer, EcDsStStagingDiskServer, EcDsSt8MMServer, EcDsStD3Server, EcDsStmgtGui, EcDmDictServer, EcDmLimServer, EcIoAdServer	This is the DBMS Interface Infrastructure Library. Sybase SQL servers operating under the DBMS, implement ECS defined guidelines for mode management, thread safety, error handling, error recovery, security, configuration, and performance of database connections.

4.8.6.17.6 Infrastructure Library Group Data Stores

Data Stores are not applicable for the Infrastructure Library Group.

4.8.6.18 Communications Subsystem Hardware CI Description

The Communications Subsystem CI is a SUN SPARC 20/712 CSS Server workstation with an external disk. Detailed specifications can be found per the site-specific hardware design diagram, baseline document number 920-TDx-001. Three DCCI software programs run on this host including the DCE Master, Domain Name Server (DNS) and Mail Server. The DCE Master consists of name, security and timeservers. The name service enables distributed applications the capability to associate information with names. The DCE Time Service (DTS) provides synchronization of all system clocks. The security service provides secure transfer of data over the network. DNS enables host names to be distinguished based on their host name and IP address relationship. The Mail Server provides standard electronic mail capability.

Detailed mappings can be found for the site-specific hardware/software is in baseline document number 920-TDx-002.

A SUN standard multi-pack is used for external storage of all components described above. A detailed configuration is specified per disk partition in baseline document number 922-TDx-005.

Other hosts and various hardware configuration items are used by the Communications Subsystem DCCI.

The Subscription Server (SBSRV), ASTER DAR Communications Gateway, ASTER E-Mail Parser Gateway and Landsat 7 Gateway run on the DMS, INTHW HWCI, Interface Server pair. (Detail specifications can be found per the site-specific hardware design diagram, baseline document number 920-TDx-001.) The SBSRV detects previously defined events. The ASTER DAR Communications Gateway provides interoperability between the DAR Client GUI tool and the DAR API. The ASTER E-Mail Parser Gateway supports automated delivery of ASTER Expedited Data Sets. Finally, the Landsat 7 Gateway provides ECS users the capability to access Landsat 7 satellite data.

A Bulletin Board Service is available at the SMC DAAC only and supported by the Bulletin Board Server. Bulletin Board messages are sent to member in a specific group.

The SMC provides two Sun servers, one primary and one (cold) secondary, to receive data from external (non-EBnet) data providers. These servers, SPARC 20/50 class systems with 18 GB of external storage, are the IGS FTP Servers.

Because the CSS software runs on multiple hosts, hardware fail-over depends on the application host.

The Subscription Server (SBSRV), ASTER DAR Communications Gateway, ASTER E-mail Parser Gateway and Landsat 7 Gateway are stored to local disk on the DMS Interface Server pair. One of the hosts is designated as the primary server as specified per the DAAC Site Host Mapping, baseline document number 910-TDA-005. In the event of failure to the primary host, a new session is initiated on the secondary host.

The CSS Server is a stand-alone host and intrinsically does not have fail-over capability. The DCE Master is replicated on a separate host to enable DCE operations to continue. DNS and DTS are loaded on multiple hosts designated as secondary. Any one of these hosts can operate as

primary servers for the DNS or DTS services in the event of non-recoverable hardware failure of the primary host.

The MSS File Server RAID is dual ported between the MSS File Server and the CM Server. In the event of a MSS File Server failure, the CM Server assumes total ownership of the RAID and all processes.

Specific primary and secondary host designations are specified per the DAAC Site Host Mapping, baseline document number 910-TDA-005.

The Bulletin Board Server is considered a non-critical function and does not have fail-over capability.

4.9 System Management Subsystem Overview

The System Management Subsystem (MSS) provides a complement of tools and services to manage ECS operations. The management services provided cover five major areas including fault, configuration, accountability, performance, and security (FCAPS). The MSS is implemented using COTS products customized to meet ECS requirements, wherever possible. The MSS maintains policy neutrality in implementing ECS management support.

The MSS software is installed locally at each DAAC to manage production operations. The MSS software is also installed at the System Management Center (SMC) at GSFC to monitor and coordinate activities involving multiple sites and to perform designated common support functions for all sites. Additionally, the ECS User Profile database is maintained at the SMC and replicated to the DAACs for their local use.

System Management Subsystem Context

Figure 4.9-1 is the System Management Subsystem context diagram. The external systems referred to in the context diagram are EDOS, ASTER, NASA Science Internet (NSI), Version 0 (V0) Information Management System (IMS), ESDIS, Science Users, and the Landsat 7 LPGS. Table 4.9-1 provides descriptions of the interface events shown in the MSS context diagram.

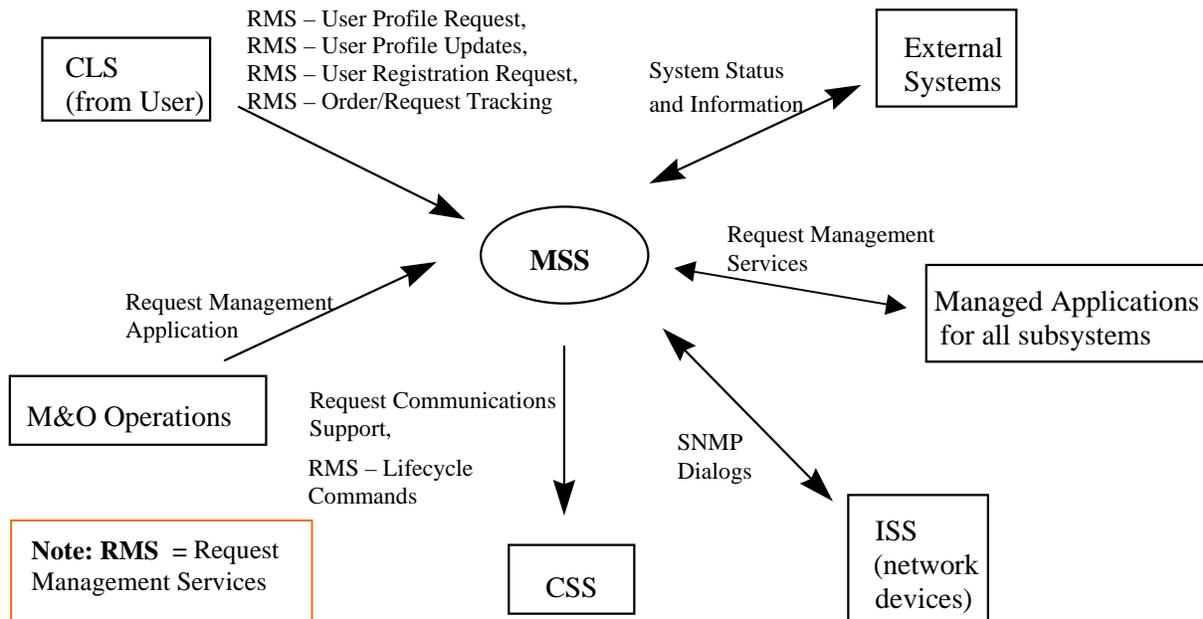


Figure 4.9-1. System Management Subsystem Context Diagram

Table 4.9-1. System Management Subsystem Interface Events (1 of 2)

Event	Interface Event Description
Request Management Services	<p>The MSS provides a basic management library of services to the subsystems, implemented as client or server applications, using the CSS Process Framework. The basic management library of services include:</p> <ul style="list-style-type: none"> • Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training). <p>The MSS also interfaces with other subsystems to perform the following:</p> <ul style="list-style-type: none"> • User Profile Request - The MSS provides requesting subsystems with access to User Profile parameters such as e-mail address and shipping address to support their processing activities. • Order/Request Tracking - The CLS uses CGI scripts to interface with the MSS Order/Request Tracking service to create a user product order and submit the order to the PLS. • User Profile Updates – The MSS receives user profile parameter updates from a user and makes the updates in the user profile database. • User Registration Request – The MSS receives user information for becoming a registered user of the ECS from the CLS. The MSS sends a response to the user when the request is received. The MSS sends the user the actual account information when the registration process is completed.
System Status and Information	<p>The MSS exchanges system status, trouble reports, and management report information with external systems such as the ASTER GDS, NSI V0 IMS, and the Landsat 7 LPGS via the EBnet.</p>
Simple Network Management Protocol (SNMP) Dialogs	<p>The MSS monitors and controls network devices such as routers and concentrators using the industry standard SNMP protocol.</p>
Request Communications Support	<p>The CSS provides a library of services available to each SDPS and CSMS subsystem. The services required to perform specific subsystem assignments are requested by the subsystem from the CSS. These services include: DCE support, file transfer services, Network and Distributed File Servers, Bulk Data transfer services, file copying services, name/address services, password services, Server Request Framework (SRF), UR, Error/Event logging, message passing, Fault Handling services, User Authentication services, Mode information and retrieving the requested configuration attribute-value pairs from the Configuration Registry for ECS applications that request them.</p>

Table 4.9-1. System Management Subsystem Interface Events (2 of 2)

Event	Interface Event Description
Request Management Application	The Maintenance and Operations (M&O) staff interact with the MSS management application service tools in the fault, configuration, accountability, performance and security management areas. These tools enable the Operations staff to collect information/metrics, schedule resources for maintenance, monitor and analyze trends, maintain the baseline and schedules, and maintain user profiles.

System Management Subsystem Structure

The MSS is comprised of three CSCIs and one hardware CI. The three CSCIs are the Management Software CSCI (MCI), the Management Agent CSCI (MACI) and the Management Logistics CSCI (MLCI). The MCI is mainly COTS software and provides distributed system management support capabilities in the fault, configuration, accountability, performance, and security service areas. The MCI custom software mainly consists of accountability software and custom extensions to COTS applications. The MACI is a combination of ECS developed and COTS software: Sub Agent, Deputy Agent, Proxy Agent, and Master Agent. The MACI communicates management requests and responses between the fault and performance management components of the MCI, which reside on the MSS Management Server host, and the various ECS managed resources (custom software, COTS software, and network devices), which reside on various hosts throughout the system. The MLCI is largely COTS software, some of which has been customized by the vendor for the ECS, which supports managing the configuration of the ECS.

The MSS hardware CI consists of a single hardware configuration item, the MHCI, provided at the SMC, the Earth Observing System Operations Center (EOC), and each DAAC. The MHCI includes an MSS management server, an MSS backup Server, an MSS application server, management workstations, and printers. The MHCI provides processing and storage support for the execution of the management applications within the MCI, MLCI, and part of the MACI.

Use of COTS in the Systems Management Subsystem

The MSS design uses COTS software to implement and provide management services as described below. Detailed explanations of the COTS software are provided in the CSC descriptions.

- RogueWave’s Tools.h++

The Tools.h++ class libraries provide strings and collections. These libraries must be installed with the MSS software for most of the MSS custom processes to run.

- RogueWave's DBTools.h++

The DBTools.h++ class libraries interact with the Sybase database Structured Query Language (SQL) server. The use of DBTools buffers the MSS processes from the relational database used. These libraries must be installed with the MSS software for most of the MSS custom processes to run.

- ICS' Builder Xcessory

The Builder Xcessory GUI builder tool modifies the displays of MSS GUIs. The Builder Xcessory tool also generates the C++ code that produces MSS GUIs at run time. No operational part of this tool is needed at run-time.

- Sybase (SQL Server)

Sybase's SQL server provides access for MSS to insert, update, and delete MSS database information. The Sybase SQL Server must be running during operations for the User Profile Server and Order Tracking Server to operate.

- Crack

Crack is a security management program that identifies user passwords that can be easily guessed. Crack enables systems administrators to force users to create passwords that are more difficult for a potential intruder to exploit.

- Anlpassword

Anlpassword is a security management program that enables system administrators to set certain rules for password creation (e.g., must be at least 8 characters long and contain a number or symbol). The anlpassword program makes it more difficult for passwords to be guessed and exploited by potential intruders.

- TCP Wrappers

TCP Wrappers is a security management program that monitors and controls user applications that connect to various network services, such as TFTP, EXEC, FTP, RSH, TELNET, RLOGIN, FINGER, and SYSTAT. The actions performed by the TCP Wrappers program are configurable, but consist of logging the remote host name and performing basic checks on the request origin.

- SATAN

SATAN is a security management program that helps systems administrators identify common networking-related security problems and reporting the problems without actually exploiting them. For each type of problem found, SATAN offers a tutorial that explains the problem and what its impact could be. The tutorial also explains what can be done about the problem: correct an error in a configuration file, install a bug fix from the vendor, use other means to restrict access, or simply disable the service.

- Tripwire

Tripwire is security management program, which is an integrity monitor. Tripwire uses several checksum/signature routines to detect file changes and monitors selected items of system-maintained information. Tripwire monitors permission, link, file size, and directory changes. It also detects file additions or deletions based on selected directories that are watched.
- ClearCase

ClearCase is a UNIX software change management application used to maintain algorithms at each DAAC.
- XRP II

XRP II is a configuration management tool that performs two basic functions within ECS: baseline management and inventory, logistics, and maintenance management.
- ACCELL

ACCELL is an integrated development system containing a relational database management system (UNIFY) used by XRP II. ACCELL must be installed in order to use XRP II.
- Networker

Networker is an application, which provides capabilities to backup and restore files or directories for all ECS hosts. Networker provides an interface for the system administrator to identify the files or directories for back up or restoring and performs the backup or restore according to specifications.
- DDTS

DDTS is a UNIX based configuration management tool to handle configuration change requests (CCRs) in the ECS system. DDTS provides the user the capability to generate, monitor, and report on ECS CCRs.
- Remedy's Action Request System (ARS)

The Remedy ARS (usually referred to as "Remedy") is a trouble ticketing application. Remedy generates, monitors, and reports on trouble tickets within the ECS. A custom ECS web interface to Remedy provides registered users the capability to generate and obtain status of the ECS trouble tickets via the web. Remedy also provides the DAAC User Services operators with a User Contact Log to maintain records of all contacts with ECS end users.
- BMC's Optima (formerly Peer's) Master Agent

The BMC Optima Master Agent (generally referred to as the Peer Agent) provides an extensible SNMP agent capability that enables HP OpenView to manage ECS custom applications. Peer agent code is built into the custom ECS agent code and is part of the ECS standard custom code delivery.

- HP OpenView

HP OpenView is a network management application extended by ECS to provide application management capabilities. The application provides a GUI for status monitoring of all ECS network devices and custom applications.

- Tivoli Software Services

Tivoli Software Services are the Tivoli Enterprise Console, Tivoli Event Server, Tivoli Sentry, Tivoli Logfile Adapter, and the Tivoli Managed Region Server (TMR) described in the Network and Enterprise Management Framework Processes Table. The Tivoli/Enterprise Console, Tivoli Sentry, and Tivoli Courier are parts of the TMR.

- IQ/Access

IQ/Access is a report generation tool to write and generate standardized management reports from the MSS database.

- Netscape Enterprise Server

The Netscape Enterprise Server implements a web interface to the Remedy Action Request System (ARS) enabling ECS users to submit trouble tickets to ARS and review the status of existing trouble tickets.

- Perl

The Perl language is used to attach and detach the ASTER standard header for e-mail sent to and received from the ASTER GDS.

- FLEXlm

FLEXlm is a license manager for controlling the use of software products. It manages license checkout and checkin processing, logs licensing events and errors, removes user licenses for specific features, displays the status of installed licenses and network licensing activities, and reports hostids of a system.

- DCE Client

DCE Client provides MSS with communications between other subsystems. DCE can reside on one or both sides of the interface. An instance must be installed on the platform where MSS resides. Although the DCE Client is part of CSS, this COTS item must be installed for MSS to run in the SDPS operational and test environment.

4.9.1 Management Software Computer Software Configuration Item Description

4.9.1.1 Management Software Functional Description

The Management Software CSCI (MCI) provides distributed system management support capabilities in the fault, configuration, accountability, performance, and security service areas. Its Computer Software Components (CSCs) include:

- **Network and Enterprise Management Framework:** This CSC enables M&O to monitor and control communications devices, hosts, and applications in the distributed system. It also provides the framework for integrating a range of other management service applications.
- **Security:** The security service is implemented using a variety of free-ware or public domain packages which monitor and evaluate the various aspects of the security setup at each site and reports status.
- **Accountability Management:** The accountability management support is provided by custom developed software for user registration and user profile attribute updates. The accountability management CSC also provides a tracking mechanism for user product orders.
- **Trouble Ticket:** The Trouble Ticket CSC manages system problem reports submitted by users and by external systems. The trouble ticket CSC also records problem assignees, tracks investigation progress, and provides users with problem resolution status.
- **Network Backup/Restore:** The Network backup and restore CSC enables the Operations staff to perform system backups and restores from a central administration position (at each site).
- **ASTER Standard Header Handler:** The ASTER standard header handler CSC supports the ECS to ASTER GDS interface and inserts a standard header for e-mail messages sent to ASTER and removes the standard header from e-mail received from the ASTER GDS. The sequentially numbered messages are logged and can be resent by M&O staff for recovery from transfer problems.

4.9.1.2 Management Software Context

Figure 4.9.1.2-1 is the Management Software CSCI (MCI) context diagram. The diagram shows the events sent to the MCI and the events the MCI sends to the other CSCIs. Table 4.9.1.2-1 provides descriptions of the interface events shown in the MCI context diagram.

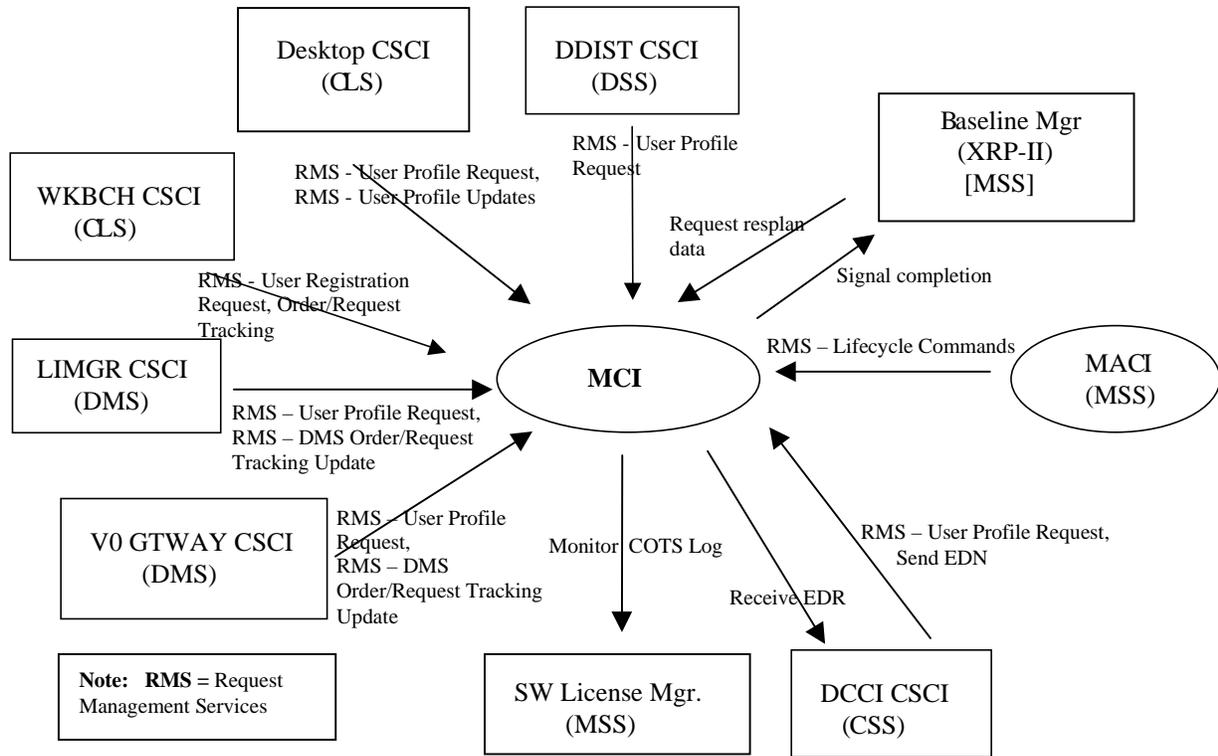


Figure 4.9.1.2-1 Management Software CSCI (MCI) Context Diagram

Table 4.9.1.2-1. Management Software CSCI (MCI) Interface Events (1 of 2)

Event	Interface Event Description
Request Management Services	<p>The MACI provides a basic management library of services to the CSCIs, implemented as client or server applications, using the DCCI CSCI Process Framework. The basic management library of services include:</p> <ul style="list-style-type: none"> • Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MACI Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MACI Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MACI Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., OPS, test, or training). <p>The MCI also interfaces with other subsystems to perform the following:</p> <ul style="list-style-type: none"> • DMS Order/Request tracking update - The LIMGR and V0 GTWAY CSCIs interface with the MCI Order/Request Tracking service to create a user product order record. • Order/Request Tracking - The CLS uses CGI scripts to interface with the MSS Order/Request Tracking service to create a user product order and submit the order to the PLS. • User Profile Request - The MCI provides requesting CSCIs with access to User Profile parameters such as e-mail address and shipping address to support their processing activities. • User Registration Request - A user submits a request to be a registered user of the ECS. Registered users are given special privileges not awarded to guests, such as the capability to order data on a media that comes at a cost. There is an immediate response to the user that the User Registration Request was received by ECS. The user receives the actual account information via the U.S. mail.

Table 4.9.1.2-1. Management Software CSCI (MCI) Interface Events (2 of 2)

Event	Interface Event Description
Request resplan data	<p>The MLCI receives a request from the MCI for data about hosts, hardware, software, and disk partitions constituting the site's production baseline as of a specified date. Arguments associated with the request are the baseline date and a code used by MCI to notify Resource Planners of the outcome of the request. The request format is:</p> <pre>resplan <mmddy> <code></pre> <p>In response, a set of ASCII records containing one header record followed by one or more detail records. The header record contains a text message identifying the production baseline specified and the number of data records. Detail records describe the items marked as "planning resource" in the Baseline Manager database that constitute the site's production baseline as of an operator-specified date. Data in a detail record is separated by a pipe symbol " " and varies by type of item as follows:</p> <ul style="list-style-type: none"> • host items - "host", name, description, control item id, status, install date, # CPUs, total RAM, processing string name, string's control item id • hardware items - "hardware", name, description, control item id, status, install date • software items - "software", description, version, control item id, status, install date, associated host name, host control item id • disk partition items - "partition", device name, directory name, control item id, status, install date, partition size, block size, logical allocation, associated host's name, host's control item id • processing string items - "string", control item id, name, description, status, install date
Signal completion	<p>A notification from the MLCI to inform the MCI that the resplan data request has been processed. The notice is made via Tivoli's "wasync" utility. It contains:</p> <ul style="list-style-type: none"> • a code indicating the purpose of the notice • an associated informational message
Send EDN	<p>The DCCI CSCI stores the EDN messages with URs, time range, etc., and sends the EDN to the MCI.</p>
Receive EDR	<p>The MCI strips the EDR message header and the message is sent to the DCCI CSCI.</p>
Monitor COTS Log	<p>The MCI periodically checks COTS log files to determine if significant events (e.g., "server died," "connection lost," or "license due to expire mm/dd/yy) have occurred. Data from identified significant events are extracted and inserted into a Tivoli event to be forwarded to the Operations Staff.</p>

4.9.1.3 Management Software Architecture

Figures 4.9.1.3-1, 4.9.1.3-2, 4.9.1.3-3, and 4.9.1.3-4 are the Management Software CSCI (MCI) architecture diagrams. The architecture diagrams show the events sent to the MCI processes and the events sent by the MCI processes to other processes or COTS software. The Operations Staff, in addition to the COTS GUIs, also invokes two custom processes (via the HPOV GUIs) to enable them to manage modes and monitor hosts for information about applications running on the host. These processes are the Mode Manager (EcMsCmModeMgr) and the ECS MIB Browser (EcMsCmMibBwsr). The Mode Manager enables the Operations Staff to manage the addition and removal of modes during operations. The ECS MIB Browser enables the Operations

Staff to query a host for information (configuration, performance, and fault) about client applications installed or running on the host.

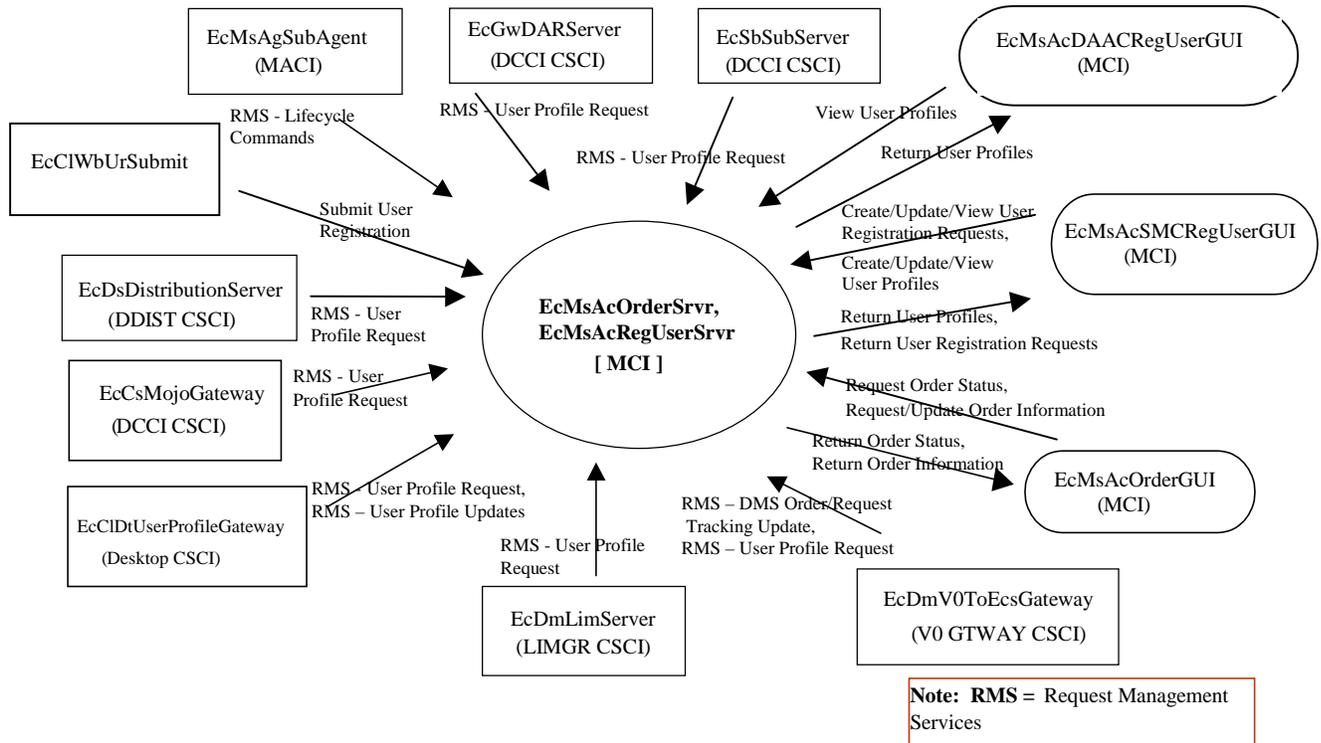


Figure 4.9.1.3-1. Management Software CSCI (MCI) Architecture Diagram

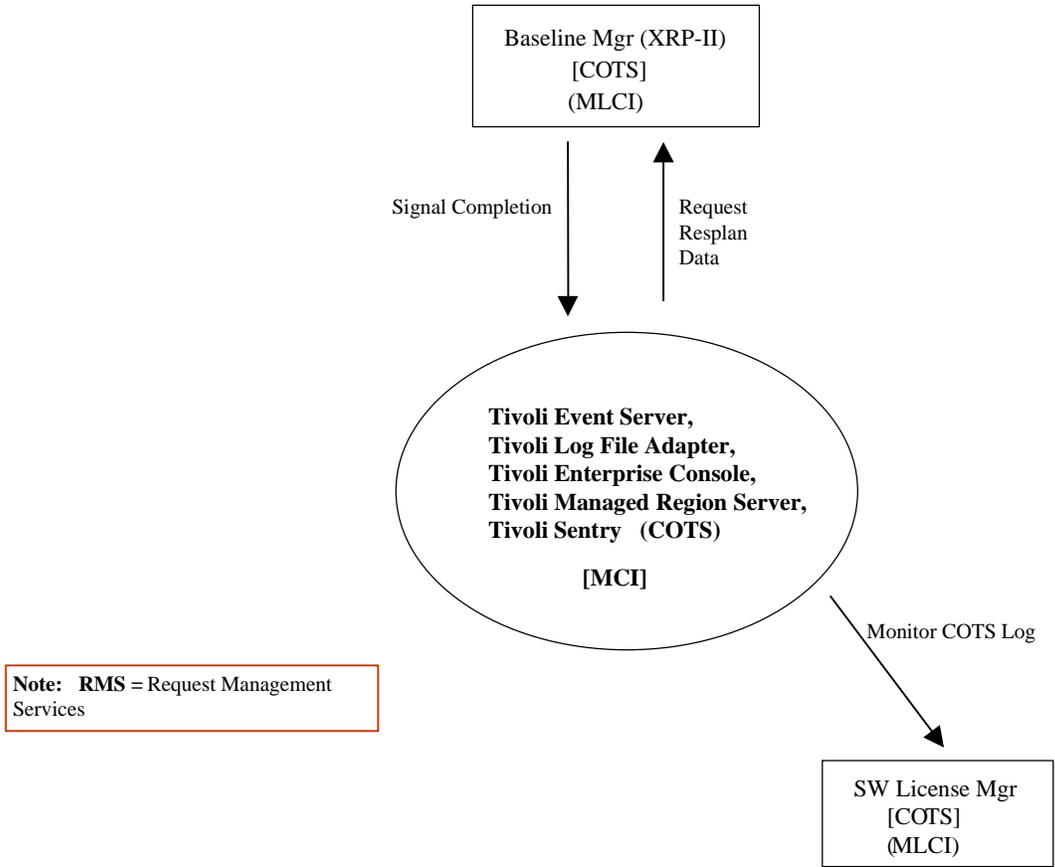


Figure 4.9.1.3-2. Management Software CSCI (MCI) Architecture Diagram (cont.)

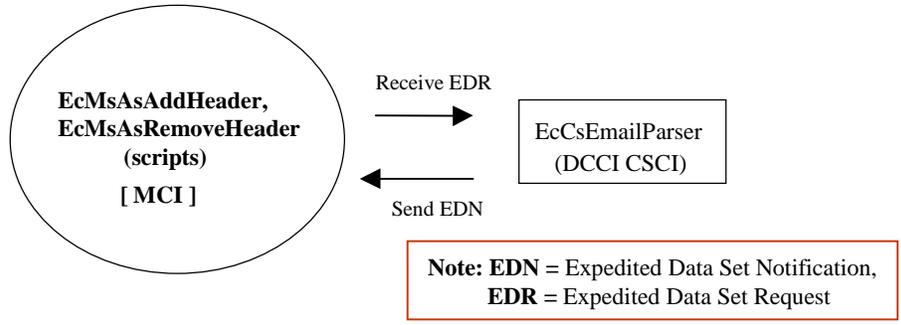


Figure 4.9.1.3-3. Management Software CSCI (MCI) Architecture Diagram (cont.)

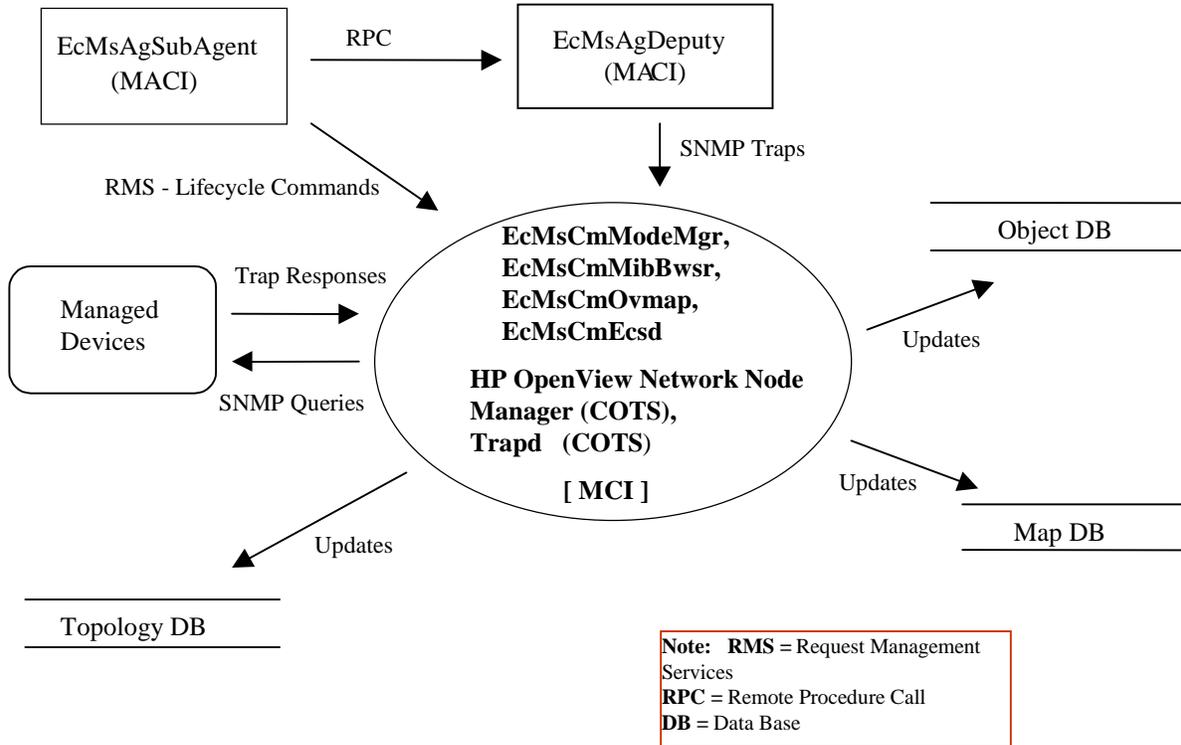


Figure 4.9.1.3-4. Management Software CSCI (MCI) Architecture Diagram (cont.)

4.9.1.4 Management Software Process Descriptions

Table 4.9.1.4-1 provides descriptions of the Management Software CSCI (MCI) processes shown in the MCI architecture diagram.

Table 4.9.1.4-1. Management Software CSCI (MCI) Processes (1 of 3)

Process	Type	COTS/ Developed	Functionality
EcMsAcRegUserSrvr	Server	Developed	<p>The User Registration Server provides an internal interface to the User Registration GUI and an external interface to other CSCIs/CSCs. The functions are:</p> <ol style="list-style-type: none"> 1. Insert, delete, update, retrieve user registration request 2. Insert, delete, update, retrieve registered user profile 3. Retrieve a list of user registration requests 4. Retrieve a list of registered user profiles 5. Change V0 gateway password <p>The EcMsAcRegUserSrvr supports:</p> <ul style="list-style-type: none"> • Single requests at a time • Multiple concurrent requests • Asynchronous request processing • Request processing de-coupled from an RPC thread (What does this mean?) • Multiple threads within a single request
EcMsAcSMCRegUserGUI	GUI	Developed	<p>The User Registration graphical user interface enables the viewing and updating of user profiles. The GUI enables the user to :</p> <ol style="list-style-type: none"> 1. Add an ECS user and send e-mail notification 2. Delete an ECS user 3. Modify an ECS user profile 4. Change the V0 gateway password 5. Change ASTER category and send e-mail 6. Change the DAR privilege <p>The ASTER e-mail address described above is stored in the Accountability configuration file. The Accountability configuration file is read when the Accountability GUI is started.</p>
EcMsAcDAACRegUserGUI	GUI	Developed	<p>The User Registration graphical user interface enables the viewing of user profiles. The GUI enables the user to list and view ECS user profiles.</p>
EcMsCmModeMgr	Other	Developed	<p>This application is launched by the Operations Staff from the OpenView GUI and enables the Operations Staff to manage the addition and removal of modes from service.</p>
EcMsCmMibBwsr	Other	Developed	<p>This application is launched by the Operations Staff from the OpenView GUI and enables the Operations Staff to query the Sub Agent for information about the ECS applications installed and/or running on that host. All responses are displayed textually.</p>

Table 4.9.1.4-1. Management Software CSCI (MCI) Processes (2 of 3)

Process	Type	COTS/ Developed	Functionality
EcMsCmOvmap	Other	Developed	This process is started when the Operations Staff starts an OpenView GUI session with the command /opt/OV/bin/ovw&. This process is responsible for ECS topology events from the event loop and populating a corresponding symbol into the submap hierarchy. For example, when an application Y started event is received for mode X, a symbol Y with a green color status appears under the mode X submap. Programs and processes for this application are rooted under the application icon.
EcMsCmEcsd	Other	Developed	This process is registered with OpenView to be started when the OpenView background daemons are started with the command /opt/OV/bin/ovstart. This process takes ECS topology change events and populates a corresponding object in the OpenView object database. Multiple symbols can be drawn for this object in more than one submap and/or map. The OpenView GUI does not have to be running for this process to do its work.
EcMsAcOrderSrvr	Server	Developed	The Order Tracking Server provides an external interface to other CSCIs/CSCs. The functions are: <ol style="list-style-type: none"> 1. Insert, delete, update, retrieve order 2. Insert, delete, update, retrieve request 3. Retrieve a list of orders 4. Retrieve a list of requests 5. Update order status 6. Update request status The EcMsAcOrderSrvr supports: <ul style="list-style-type: none"> • Single requests at a time • Multiple concurrent requests • Asynchronous request processing • Multiple threads within a single request
HP OpenView Network Node Manager	Server	COTS	The HP OpenView Network Node Manager is several servers and a GUI. These servers maintain information about network devices and custom applications. The GUI provides an interface to allow the M&O staff: <ul style="list-style-type: none"> • to manage the device information • to control custom applications
Tivoli Event Server	Server	COTS	The Tivoli Event Server receives the events sent by Tivoli managed hosts and stores them in a proprietary database. (The database is proprietary, but is based largely on an earlier version of Sybase)

Table 4.9.1.4-1. Management Software CSCI (MCI) Processes (3 of 3)

Process	Type	COTS/ Developed	Functionality
Tivoli Log File Adapter	Other	COTS	The Tivoli Logfile Adapter resides on managed hosts and monitors log files for predetermined strings. By default, the System Log (SYSLOG) file is monitored. The adapter is also configured to monitor select COTS logs.
EcMsAsAddHeader	Other	Developed	This script is invoked by the sendmail daemon when a message is sent to an alias configured to process messages requiring ASTER e-mail headers before delivery. This perl script inserts the header into a message directed to the ASTER GDS.
EcMsAsRemoveHeader	Other	Developed	This script is also invoked by the sendmail daemon when a message is sent to an alias configured to process e-mail containing ASTER e-mail headers. The perl script removes the header and forwards the message to the address defined in the alias.

4.9.1.5 Management Software Process Interface Descriptions

Tables 4.9.1.5-1, 4.9.1.5-2, 4.9.1.5-3, and 4.9.1.5-4 provide descriptions of the interface events shown in the Management Software CSCI (MCI) architecture diagrams, respectively.

**Table 4.9.1.5-1. Management Software CSCI (MCI) Process Interface Events
(1 of 6)**

Event	Event Frequency	Interface	Initiated By	Event Description
Request Management Services (RMS)				The EcMsAgSubAgent, EcMsAcOrderSrvr and EcMsAcRegUserSrvr provide a basic management library of services to the processes, implemented as client or server applications, using the DCCI CSCI Process Framework. The basic management library of services include:
(RMS – cont.)		<i>Processes:</i> EcMsAcOrderSrvr, EcMsAcRegUserSrvr	<i>Process:</i> EcMsAgSubAgent <i>Library:</i> EcAgInstrm <i>Class:</i> EcAgManager	<ul style="list-style-type: none"> Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training).

**Table 4.9.1.5-1. Management Software CSCI (MCI) Process Interface Events
(2 of 6)**

Event	Event Frequency	Interface	Initiated By	Event Description
(RMS – cont.)		<i>Process:</i> EcMsAcOrderSrvr <i>Library:</i> MsAcCInt <i>Class:</i> EcAcOrderCMgr	<i>Process:</i> EcDmLimServer <i>Library:</i> DmLmReqProc <i>Class:</i> DmLmProductPlan <i>Process:</i> EcDmV0ToEcsGateway <i>Class:</i> DmGwRequestReceiver	The EcMsAcRegUserSrvr also interfaces with other processes to perform the following: <ul style="list-style-type: none"> • DMS Order/Request Tracking Update - The EcDmLimServer and EcDmV0ToEcsGateway interface with the EcMsAcOrderSrvr (Order/Request Tracking service) to create a user product order record.

**Table 4.9.1.5-1. Management Software CSCI (MCI) Process Interface Events
(3 of 6)**

Event	Event Frequency	Interface	Initiated By	Event Description
(RMS – cont.)		<i>Process:</i> EcMsAcRegUserSrvr, <i>Libraries:</i> MsAcCInt, MsAcComm <i>Class:</i> EcAcProfileMgr	CSS Process: EcGwDARServer <i>Class:</i> EcGwDARGatewayRequest_S CSS Process: EcSbSubServer DMS Process: EcDmV0ToEcsGateway <i>Class:</i> DmGwRequestReceiver DMS Process: EcDmLimServer <i>Library:</i> DmLmReqProc <i>Class:</i> DmLmProductPlan CLS Process: EcCIDtUserProfileGateway <i>Class:</i> CIDtUserProfileServer CSS Process: EcCsMojoGateway <i>Library:</i> EcCsMojoGateway <i>Class:</i> EcMjRetrieveProfileProxy DSS Process: EcDsDistributionServer <i>Library:</i> DsDdSSh <i>Class:</i> DsDdMedia	<ul style="list-style-type: none"> • User Profile Request - The EcMsAcRegUserSrvr provides requesting processes with user profile parameters such as e-mail address and shipping address to support their processing activities.

**Table 4.9.1.5-1. Management Software CSCI (MCI) Process Interface Events
(4 of 6)**

Event	Event Frequency	Interface	Initiated By	Event Description
(RMS – cont.)		<i>Process:</i> EcMsAcRegUserSrvr <i>Libraries:</i> MsAcCInt, MsAcComm <i>Class:</i> EcAcProfileMgr	<i>Script:</i> EcCIWbUrSubmit	<ul style="list-style-type: none"> • User Registration Request - A user submits a request to be a registered user of the ECS. Registered users are given special privileges not awarded to guests, such as the capability to order data on a media at a cost. There is an immediate response to the user that the User Registration Request was received by the ECS. The user receives the actual account information once the account is created via the U.S. mail.
View User Profiles	One per view user profile request	<i>Process:</i> EcMsAcRegUserSrvr <i>Libraries:</i> MsAcCInt, MsAcComm	M&O staff <i>Process:</i> EcMsAcDAACRegUserSrvr <i>Libraries:</i> MsAcCInt, MsAcComm	The M&O staff request, via the EcMsAcDAACRegUserGUI, to retrieve a user profile from the EcMsAcRegUserSrvr.
Return User Profiles	One per return of user profile	EcMsAcSMCRegUserGUI, EcMsAcDAACRegUserGUI <i>Libraries:</i> MsAcCInt, MsAcComm	<i>Process:</i> EcMsAcRegUserSrvr <i>Libraries:</i> MsAcCInt, MsAcComm	The EcMsAcRegUserSrvr returns user profile information provided from the Sybase Server to the requester at the EcMsAcSMCRegUserGUI or the EcMsAcDAACRegUserGUI.

**Table 4.9.1.5-1. Management Software CSCI (MCI) Process Interface Events
(5 of 6)**

Event	Event Frequency	Interface	Initiated By	Event Description
Create/Update/View User Registration Requests	One per create/update user registration	<i>Process:</i> EcMsAcRegUserSrvr <i>Libraries:</i> MsAcClnt, MsAcComm	M&O staff <i>Process:</i> EcMsAcRegUserGUI <i>Libraries:</i> MsAcClnt, MsAcComm <i>Class:</i> MsAcRegUserUI	The M&O staff send requests to create or modify user registration information via the EcMsAcRegUserGUI.
Create/Update/View User Profiles	One per create/update user profile	<i>Process:</i> EcMsAcRegUserSrvr <i>Libraries:</i> MsAcClnt, MsAcComm	M&O staff <i>Process:</i> EcMsAcSMCRegUserGUI <i>Libraries:</i> MsAcClnt, MsAcComm <i>Class:</i> MsAcRegUserUI	The M&O staff send requests to create, modify, or review user profile information via the EcMsAcSMCRegUserGUI.
Return User Profiles	One per return of user profile	<i>Processes:</i> EcMsAcSMCRegUserGUI, EcMsAcDAACRegUserGUI <i>Libraries:</i> MsAcClnt, MsAcComm	<i>Process:</i> EcMsAcRegUserSrvr <i>Libraries:</i> MsAcClnt, MsAcComm	The EcMsAcRegUserSrvr returns user profile information provided from the Sybase Server to the requester at the EcMsAcSMCRegUserGUI or the EcMsAcDAACRegUserGUI.
Return User Registration Requests	One per return of user registration requests	<i>Process:</i> EcMsAcSMCRegUserGUI <i>Libraries:</i> MsAcClnt, MsAcComm <i>Class:</i> MsAcRegUserUI	<i>Process:</i> EcMsAcRegUserSrvr <i>Libraries:</i> MsAcClnt, MsAcComm	The EcMsAcRegUserSrvr returns user registration request information provided from the Sybase Server to the requester at the EcMsAcSMCRegUserGUI.

**Table 4.9.1.5-1. Management Software CSCI (MCI) Process Interface Events
(6 of 6)**

Event	Event Frequency	Interface	Initiated By	Event Description
Request Order Status	One per request order status	<i>Process:</i> EcMsAcOrderSrvr <i>Libraries:</i> MsAcClnt, MsAcComm	M&O Staff <i>Process:</i> EcMsAcOrderGUI <i>Libraries:</i> MsAcClnt, MsAcComm	The EcMsAcOrderGUI is used (by the M&O Staff) to retrieve the current order status. The EcMsAcOrderGUI submits a request for current order status to the EcMsAcOrderSrvr.
Request/Update Order Information	One per request/update order information	<i>Process:</i> EcMsAcOrderSrvr <i>Libraries:</i> MsAcClnt, MsAcComm	M&O Staff <i>Process:</i> EcMsAcOrderGUI <i>Libraries:</i> MsAcClnt, MsAcComm	The EcMsAcOrderGUI is used (by the M&O Staff) to retrieve order information by submitting requests to the EcMsAcOrderSrvr.
Return Order Status	One per return order status	<i>Process:</i> EcMsAcOrderGUI <i>Libraries:</i> MsAcClnt, MsAcComm	<i>Process:</i> EcMsAcOrderSrvr <i>Libraries:</i> MsAcClnt, MsAcComm	The EcMsAcOrderSrvr receives the product order status from the Sybase Server and returns the status to the EcMsAcOrderGUI.
Return Order Information	One per return of order information	<i>Process:</i> EcMsAcOrderGUI <i>Libraries:</i> MsAcClnt, MsAcComm	<i>Process:</i> EcMsAcOrderSrvr <i>Libraries:</i> MsAcClnt, MsAcComm	The EcMsAcOrderSrvr receives the product order information from the Sybase Server and returns the order information to the EcMsAcOrderGUI.
Submit User Registration	One per registration request to MSS	<i>Process:</i> EcMsAcRegUserSrvr <i>Libraries:</i> MsAcClnt, MsAcComm <i>Classes:</i> EcAcUsrRequestMgr, MsAcUsrRequest	<i>Script:</i> EcCIWbUrSubmit	The URT CGI script takes the confirmed user inputs and submits the registration request to the EcMsAcRegUserSrvr at the SMC.

**Table 4.9.1.5-2. Management Software CSCI (MCI) Process Interface Events
(1 of 2)**

Event	Event Frequency	Interface	Initiated By	Event Description
Request Resplan Data	One per resplan data request	<i>Script:</i> Resplan (XRP-II)	<i>Process:</i> Tivoli Event Server (COTS)	<p>Request to XRP-II for data about hosts, hardware, software, and disk partitions constituting the site's production baseline as of a specified date. Arguments associated with the request are the baseline date and a code used by MCI to notify Resource Planners of the outcome of the request. The request format is:</p> <pre>resplan <mmddyy> <code></pre> <p>In response, a set of ASCII records containing one header record followed by one or more detail records. The header record contains a text message identifying the production baseline that was specified and the number of data records. Detail records describe the items marked as "planning resource" in the Baseline Manager database that constitute the site's production baseline as of an operator-specified date. Data in a detailed record is separated by a pipe symbol " " and varies by type of item as follows:</p> <ul style="list-style-type: none"> • host items - "host", name, description, control item id, status, install date, # CPUs, total RAM, processing string name, string's control item id • hardware items - "hardware", name, description, control item id, status, install date • software items - "software", description, version, control item id, status, install date, associated host name, host control item id • disk partition items - "partition", device name, directory name, control item id, status, install date, partition size, block size, logical allocation, associated host's name, host's control item id • processing string items - "string", control item id, name, description, status, install date

**Table 4.9.1.5-2. Management Software CSCI (MCI) Process Interface Events
(2 of 2)**

Event	Event Frequency	Interface	Initiated By	Event Description
Monitor COTS Log	Configurable (default set to every 60 seconds)	Tivoli Log File Adapter (COTS)	Tivoli Event Server (COTS)	The Tivoli Log File Adapter checks configured COTS logs at predetermined intervals. New events logged since the last check are screened and Tivoli events are created wherever new significant events (e.g., "server died," "connection lost," or "license due to expire mm/dd/yy") are found in the log file.
Signal Completion	One per resplan data request	wasync utility program	<i>Program:</i> Baseline Manager (XRP-II) [COTS]	Notification for MCI that the resplan data request has been processed. The notice is made via Tivoli's "wasync" utility. It contains: <ul style="list-style-type: none"> • a code indicating the purpose of the notice • an associated informational message.

Table 4.9.1.5-3. Management Software CSCI (MCI) Process Interface Events

Event	Event Frequency	Interface	Initiated By	Event Description
Receive EDR	One per EDR send	<i>Process:</i> EcCsEmailParser	<i>Script:</i> EcMsAsRemoveHeader	After selecting the EDN, the ASTER GDS personnel send an EDR to the EcCsEmailParser to be forwarded via the EcMsAsRemoveHeader script in the MCI after the header has been removed.
Send EDN	One per E-mail send	<i>Script:</i> EcMsAsAddHeader	<i>Process:</i> EcCsEmailParser <i>Class:</i> EcCsEmailParser	The EcCsEmailParser sends the Expedited Data Set Notification (EDN) information to the EcMsAsAddHeader script in the MCI to have a header added.

**Table 4.9.1.5-4. Management Software CSCI (MCI) Process Interface Events
(1 of 2)**

Event	Event Frequency	Interface	Initiated By	Event Description
SNMP Traps	One per trap state change	snmp / Trap protocols <i>Process:</i> Trapd	Process: EcMsAgDeputy Library: Ovsnmp Classes: MsAgDeputy, MsAgSnmpPdu, MsAgEventEntry	SNMP Traps are sent by the EcMsAgDeputy notifying of a change in the object state upon receiving information from the EcMsAgSubAgent. The information is used to update the status of objects in OpenView databases via the Trapd, EcMsCmEcsd, and EcMsCmOvmap processes.
Updates	One per update	<i>Databases:</i> OV Object, Map, Topology	<i>Processes:</i> EcMsCmEcsd, EcMsCmOvmap, HPOV (COTS) Library: Ovw Ecsd Classes: MsCmTopoEvent, MsCmMgrBkg Ovmap Classes: MsCmMgr, MsCmEvent, MsCmLookup, MsCmCtrl	The OV Object DB, Topology DB, and Map DB are updated via the HPOV API and the EcMsCmEcsd and EcMsCmOvmap processes.
SNMP Queries	One per Snmp query	snmp "gets"	Process: HPOV (COTS)	HP OpenView sends SNMP queries to managed devices and receives query responses from the managed devices to update the status of objects in OpenView (OV) databases.
Trap Responses	One per response	HPOV (COTS)	Managed Devices	SNMP query responses from managed objects. The RPC is between the OpenView management station and the subagents on each host.
Request Management Services (RMS)				The EcMsAgSubAgent provides a basic management library of services to the processes, implemented as client or server applications, using the DCCI CSCI Process Framework. The basic management library of services include:

**Table 4.9.1.5-4. Management Software CSCI (MCI) Process Interface Events
(2 of 2)**

Event	Event Frequency	Interface	Initiated By	Event Description
(RMS – cont.)		<i>Processes:</i> EcMsCmModeMgr, EcMsCmMibBwsr, EcMsCmOvmap, EcMsCmEcsd, HP Open View Network Node Manager, Trapd	<i>Process:</i> EcMsAgSubAgent <i>Library:</i> EcAgInstrm <i>Class:</i> EcAgManager	<ul style="list-style-type: none"> Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has a mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training).
RPC	One per response	<i>Process:</i> EcMsAgDeputy <i>Library:</i> EcMsAgDeputyClient	<i>Process:</i> EcMsAgSubAgent <i>Library:</i> EcMsAgDeputyClient <i>Classes:</i> MsAgEventManager, EcAgEvent	The EcMsAgSubAgent concurrently polls the process and RPC status of all ECS applications on the managed host. Changes of state are relayed to the EcMsAgDeputy via RPC events. The status is reflected in the HPOV submap hierarchies. ECS applications transmit configuration and fault events to the EcMsAgSubAgent, which relays the events to the EcMsAgDeputy for forwarding to HP OpenView. HP OpenView updates its configuration and fault logs for viewing by the Operations Staff.

4.9.1.6 Management Software Data Stores

Data stores are not applicable for the Management Software CSCI.

4.9.1.6.1 MCI - Network and Enterprise Management Computer Software Component Description

The Management Software CSCI (MCI) is COTS and custom software enabling the Operations staff to monitor and coordinate the ECS services. The MCI is the following CSCs:

1. Network and Enterprise Management Framework
2. Security Service
3. Accountability Management
4. Trouble Ticket
5. Network Backup/Restore
6. ASTER E-mail Header Handler

4.9.1.6.1.1 Network and Enterprise Management Framework Functional Overview

The network and enterprise management framework monitors and controls the network, applications, and hosts distributed throughout the network. The framework is made of the HP OpenView Network Node Manager (NNM) and the Tivoli Enterprise Console (TEC) COTS products. The OpenView NNM and the Tivoli Enterprise Console monitor and control the network and are the integration platform for other management tools, both custom and COTS, for a range of system management needs (e.g., network management, GUI development, and database usage).

Network Management Framework - OpenView

OpenView Network Node Manager (NNM) uses the Simple Network Management Protocol (SNMP) to monitor and control network objects. OpenView NNM has a “discovery” service that automatically detects network devices such as routers, bridges, and hosts and adds these devices to its database. Identification of other ECS managed objects is provided through ECS developed software in the Management Agent CSCI (MACI). The MACI informs the OpenView NNM of application elements. This information, along with mode management information, is collected by the OpenView NNM, saved in the map, object, and topology data views (tables) within the OpenView Management Database (DB) (in the Map DB, Object DB, and Topology DB views (tables)). This information is also used to build operations maps to show the logical layout and status of ECS managed objects. OpenView NNM also provides the capability to develop scripts that define action routines for each event received from each managed object (a device). OpenView NNM provides API support for integrating other management application packages and for developing custom management applications.

Enterprise Management Framework - Tivoli Enterprise Console

Tivoli monitors the performance of all managed hosts. The Tivoli Event server receives performance information from the Tivoli Sentry agent residing in each managed host, including

free disk space, amount of swap space available, CPU usage, and number of active processes. When any of these metrics exceed a configurable threshold set by the Operations staff, Sentry sends the warning to the management station notifying the Operations staff of the potential system impact. Tivoli also supports the process scheduling on hosts in its management region to satisfy routine administrative tasks.

Fault Management Service

OpenView NNM and Tivoli provide basic fault management tools. The OpenView NNM receives general fault notifications from managed network devices via SNMP traps. Tivoli receives host performance threshold notifications when thresholds have exceeded an unacceptable level. The framework tools possess a rules-based capability for reacting to faults, updating managed object operational status, and forwarding fault notices to the SMC.

Performance Management Service

Performance management is a task performed by the Operations Staff using tools provided by the Network and Enterprise Framework. The OpenView NNM and Tivoli framework enable the Operations Staff to collect and display resource usage trend information. The NNM is used to monitor network device performance parameters such as packet throughput. The Tivoli monitors host resource usage thresholds and warns operations when they are exceeded.

Mode Management Service

The Mode Management Service (MMS) is an ECS developed service that is tightly integrated with HP OpenView. The MMS enables ECS applications to be configured into an operational mode and also provides support for ECS applications to be configured into training and testing modes during operations. The MMS incorporates the mode management user interface directly into the HP OpenView GUI and provides methods to activate and deactivate a mode. In addition, the MMS provides a mode specific user interface for accessing CSS lifecycle control (start-up and shutdown) commands. Monitoring capabilities are provided within HP OpenView and are enhanced to reflect mode specific status of software system, subsystem, application, program, and process level entities. Hardware is mode independent so its status is reflected outside of the mode structure. HP OpenView graphically supports multiple modes through the use of separate sub-maps and symbol labels. The map can have any number of sub-maps defined that decompose the basic high-level map representation. Each mode has its mode specific map (and associated sub-maps) predefined to recognize and support the software items within the given mode.

4.9.1.6.1.2 Network and Enterprise Management Framework Context

Figure 4.9.1.6.1.2-1 is the Network and Enterprise Management Framework context diagram. The diagram shows the events sent to the Network and Enterprise Management Framework CSC and the events the Network and Enterprise Management Framework CSC sends to other CSCIs or CSCs. Table 4.9.1.6.1.2-1 provides descriptions of the interface events shown in the Network and Enterprise Management Framework context diagram.

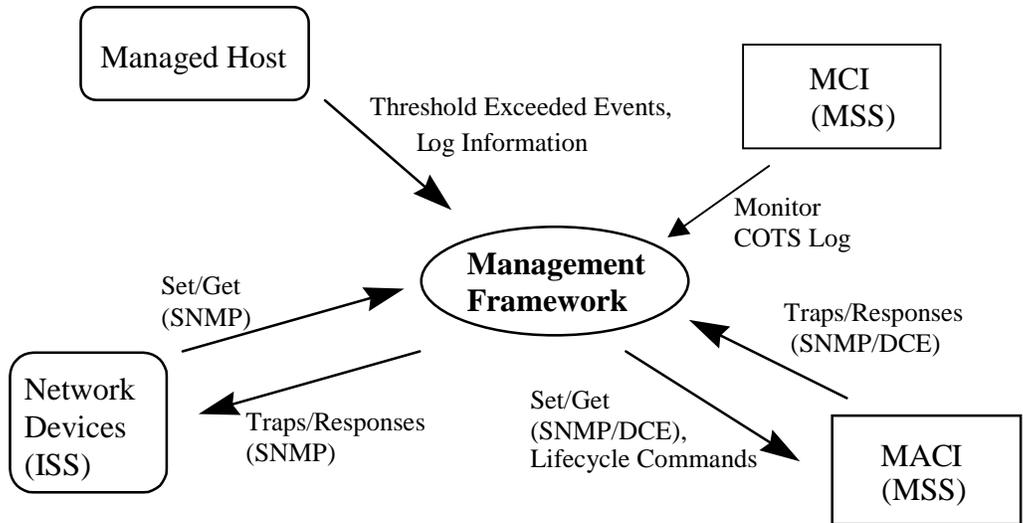


Figure 4.9.1.6.1.2-1. Network and Enterprise Management Framework Context Diagram

Table 4.9.1.6.1.2-1. Network and Enterprise Management Framework Interface Events

Event	Interface Event Description
Monitor COTS Log	The MCI periodically checks COTS log files to determine if significant events (e.g., "server died," "connection lost," or "license due to expire mm/dd/yy) have occurred. Data from identified significant events are extracted and inserted into a Tivoli event to be forwarded to the Operations Staff.
Traps/Responses (SNMP/DCE)	The MSS management agent (MACI) in each managed object sends secure emulations via DCE Remote Procedure Calls for responses sent by managed applications.
Set/Get (SNMP/DCE)	The MACI in each managed host receives secure emulations of set/get commands via DCE Remote Procedure Calls for commands sent to managed applications.
Lifecycle Commands	HP OpenView Network Node Manager issues startup/shutdown lifecycle commands via the MACI to applications in the managed hosts.
Traps/Responses (SNMP)	The SNMP traps/responses used by network snmp compliant devices to send event notifications and requested MIB variable responses to the OpenView NNM.
Set/Get (SNMP)	The SNMP set/get commands used by HP OpenView with network devices to set MIB variables and to get the value of a MIB variable, respectively.
Threshold Exceeded Events	Managed hosts with the Tivoli Sentry agent residing in them provide host resource usage metrics to the Tivoli Enterprise Console.
Log Information	Managed hosts with the Tivoli Sentry agent provide access to host log information.

4.9.1.6.1.3 Network and Enterprise Management Framework Process Architecture

Figures 4.9.1.6.1.3-1 and 4.9.1.6.1.3-2 are the Network and Enterprise Management Framework architecture diagrams. The diagrams show the events sent to the Network and Enterprise Framework CSC processes and the events the Network and Enterprise Framework processes send to other processes.

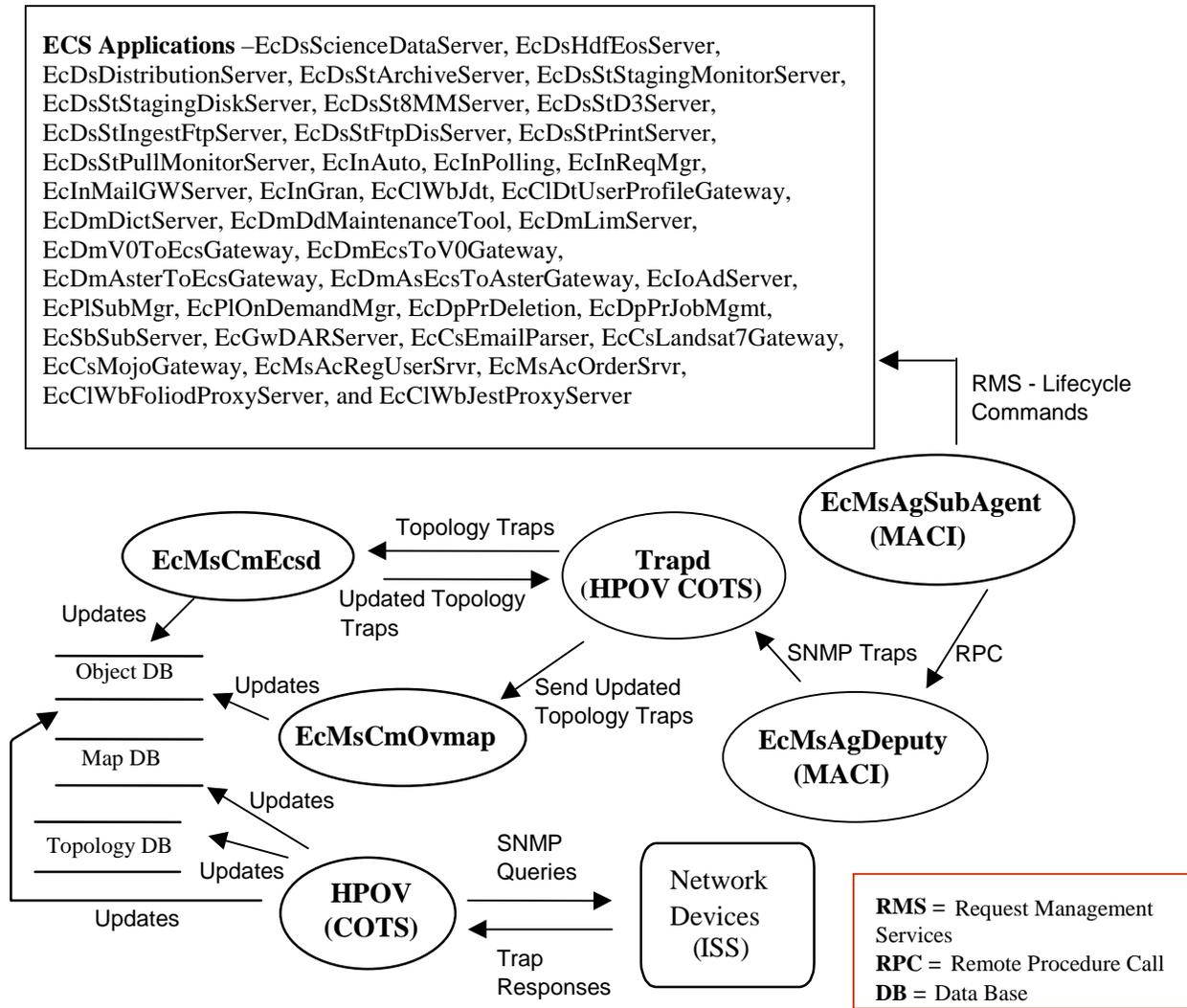


Figure 4.9.1.6.1.3-1. Network and Enterprise Management Framework Architecture Diagram

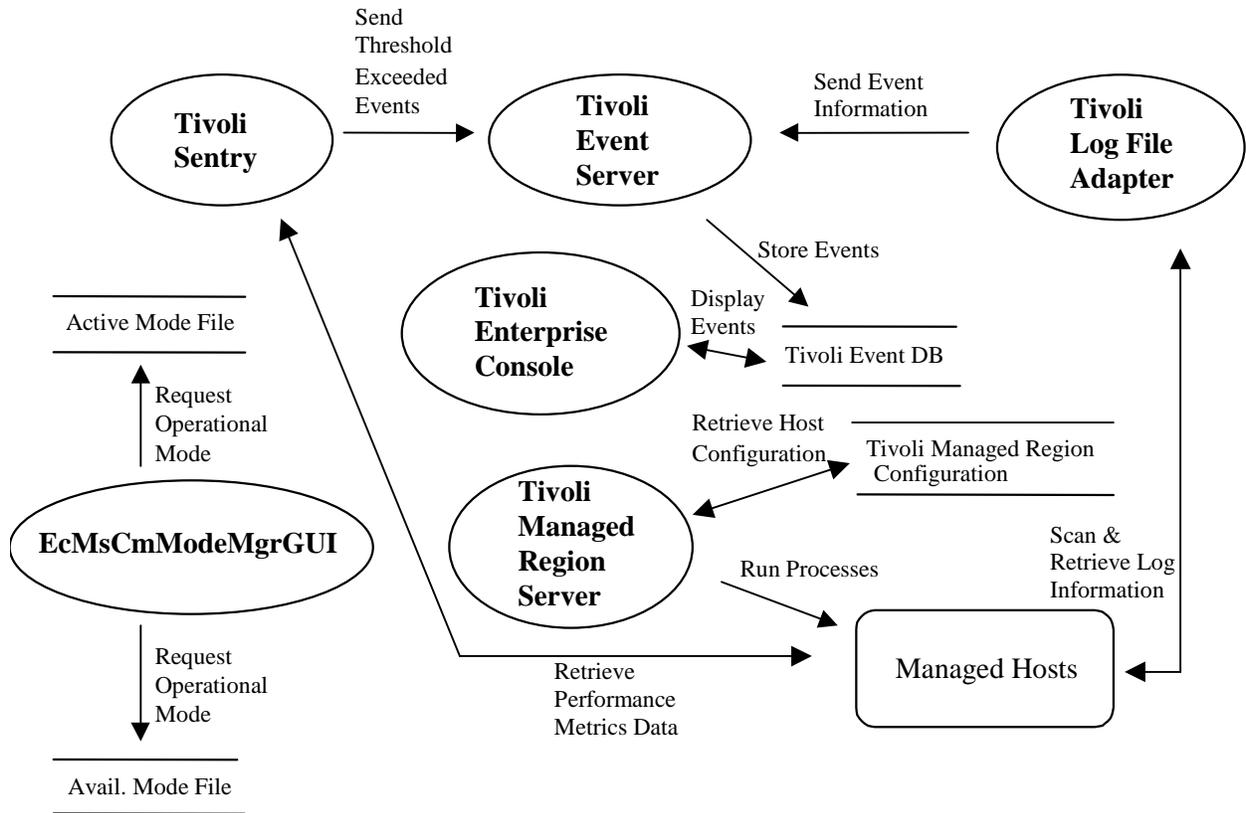


Figure 4.9.1.6.1.3-2. Network and Enterprise Management Framework Architecture Diagram (cont.)

4.9.1.6.1.4 Network and Enterprise Management Framework Process Descriptions

Table 4.9.1.6.1.4-1 describes the processes in the Network and Enterprise Management Framework architecture diagram.

Table 4.9.1.6.1.4-1. Network and Enterprise Management Framework Processes (1 of 3)

Process	Type	COTS/ Developed	Functionality
HPOV	Daemon	COTS	The HPOV process sends SNMP queries to managed objects in the network (devices) and uses the response information to update the OV Object DB and Topology DB. This process can also manipulate the MIB variable settings using the SNMP Set command.

**Table 4.9.1.6.1.4-1. Network and Enterprise Management Framework Processes
(2 of 3)**

Process	Type	COTS/ Developed	Functionality
EcMsCmOvmap	Other	Developed	This process is started when the Operations Staff starts an OpenView GUI session with the command /opt/OV/bin/ovw&. This process is responsible for extracting ECS topology events from the event loop and populating a corresponding symbol into the submap hierarchy. For example, when an application Y started event is received for mode X, a symbol Y with a green color status appears under the mode X submap. Programs and processes for this application are rooted under the application icon.
EcMsCmEcsd	Other	Developed	This process is registered with OpenView to be started when the OpenView background daemons are started with the command /opt/OV/bin/ovstart. This process takes ECS topology change events and populates a corresponding object in the OpenView object database. Multiple symbols can be drawn for this object in more than one submap and/or map. The OpenView GUI does not have to be running for this process to do its work.
Trapd	Daemon	COTS	The Trapd process receives SNMP traps from SNMP daemons running on each host and from the EcMsAgDeputy. Other applications, such as EcMsCmEcsd and EcMsCmOvmap, register with Trapd to be notified whenever certain types of traps are received.
Tivoli Event Server	Server	COTS	The Tivoli Event Server receives the events sent by Tivoli managed hosts and stores them in a Sybase database.
Tivoli Sentry	Server	COTS	The Tivoli Sentry resides on each managed host and monitors system resource usage. Tivoli Sentry sends threshold exceeded events to the Tivoli Event Server whenever usage goes above a configurable level.
Tivoli Enterprise Console (TEC)	GUI	COTS	The TEC provides the user with notifications if a threshold is exceeded. Examples include: CPU usage exceeding 97% - Critical Event Disk usage exceeds 95% - Critical Event Swap Space available is below 10 Megabytes (MB) – Warning Event The AutoSys daemon has become unavailable – Critical The string “REPEATED LOGIN FAILURES” was found in the syslog – WARNING Event
Tivoli Log File Adapter	Other	COTS	The Tivoli Logfile Adapter resides on managed hosts and monitors log files for predetermined strings. By default, the System Log (SYSLOG) file is monitored. The adapter is also configured to monitor select COTS logs.

**Table 4.9.1.6.1.4-1. Network and Enterprise Management Framework Processes
(3 of 3)**

Process	Type	COTS/ Developed	Functionality
Tivoli Managed Region (TMR) Server	Server	COTS	The Tivoli Managed Region Server resides at the management station. The Tivoli Managed Region Server communicates with client hosts in the Tivoli management region. The server maintains the status of these hosts and is capable of scheduling the running of scripts on these hosts for routine administration.
EcMsCmModeMgr GUI	GUI	Developed	The Mode Management GUI provides a means for operations to define modes and initiate and control applications in all of the defined modes.
EcMsAgDeputy	Other	Developed	The EcMsAgDeputy converts the application events received from the EcMsAgSubAgent and converts them into traps. The traps are forwarded to the Trapd daemon, which logs the traps into the trapd log where they are read by HPOV.
EcMsAgSubAgent	Other	Developed	The EcMsAgSubAgent sends the lifecycle commands (startup and shutdown) to the managed ECS applications and sends application events (in the form of Remote Procedure Calls (RPCs)) to the EcMsAgDeputy.

4.9.1.6.1.5 Network and Enterprise Management Framework Process Interface Descriptions

Table 4.9.1.6.1.5-1 provides descriptions of the Network and Enterprise Management Framework interface events shown in the Network and Enterprise Management Framework architecture diagram.

Table 4.9.1.6.1.5-1. Network and Enterprise Management Framework Process Interface Events (1 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
Request Management Services	One command per request	<p><i>Processes:</i> EcDsScienceDataServer, EcDsHdfEosServer, EcDsDistributionServer, EcDsStArchiveServer, EcDsStStagingMonitorServer, EcDsStStagingDiskServer, EcDsSt8MMServer, EcDsStD3Server, EcDsStIngestFtpServer, EcDsStFtpDisServer, EcDsStPrintServer, EcDsStPullMonitorServer, EcInAuto, EcInPolling, EcInReqMgr, EcInEmailGWServer, EcInGran, EcCIWbJdt, EcCIDtUserProfileGateway, EcDmDictServer, EcDmLimServer, EcDmV0ToEcsGateway, EcDmEcsToV0Gateway, EcDmAsterToEcsGateway, EcDmEcsToAsterGateway, EcIoAdServer, EcPISubMgr, EcPIOdMgr, EcDpPrDeletion, EcDpPrJobMgmt, EcSbSubServer, EcGwDARServer, EcCsEmailParser, EcCsLandsat7Gateway, EcCsMojoGateway, EcMsAcRegUserSrvr, EcMsAcOrderSrvr, EcCIWbFoliodProxyServer, EcCIWbJessProxyServer</p>	<p><i>Process:</i> EcMsAgSubAgent</p> <p><i>Library:</i> EcAgInstrm</p> <p><i>Class:</i> EcAgManager</p>	<p>The EcMsAgSubAgent provides a basic management library of services to the processes, implemented as client or server applications, using the DCCI CSCI Process Framework. The basic management library of services include:</p> <ul style="list-style-type: none"> <p>Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training).</p>

Table 4.9.1.6.1.5-1. Network and Enterprise Management Framework Process Interface Events (2 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
RPC	One per response	<i>Process:</i> EcMsAgDeputy <i>Classes:</i> MsAgDeputyGate, MsAgSnmpPdu	<i>Process:</i> EcMsAgSubAgent <i>Library:</i> DCE <i>Classes:</i> MsAgEventMgr, EcAgEvent	ECS applications communicate with the EcMsAgSubAgent, which passes information to the EcMsAgDeputy to be sent to HP OpenView to update databases about the status of managed applications.
SNMP Traps	One per trap state change	snmp / Trap protocols <i>Process:</i> Trapd	<i>Process:</i> EcMsAgDeputy <i>Libraries:</i> Ovw, Ovsnmp	SNMP Traps are sent by the EcMsAgDeputy notifying of a change in the object state upon receiving information from the EcMsAgSubAgent. The information is used to update the status of objects in OpenView databases via the Trapd, EcMsCmEcsd, and EcMsCmOvmap processes.
Send Updated Topology Traps	One per topology state change	<i>Process:</i> EcMsCmOvmap <i>Library:</i> Ovw	<i>Process:</i> Trapd (COTS)	The updated topology trap is sent to the EcMsCmOvmap so the OpenView (OV) map can be updated.
SNMP Queries	One per Snmp query	Managed Network Devices	<i>Process:</i> HPOV (COTS) [snmp "gets"]	HP OpenView sends SNMP queries to managed devices and receives trap responses from the managed devices to update the status of objects in OpenView (OV) databases.

Table 4.9.1.6.1.5-1. Network and Enterprise Management Framework Process Interface Events (3 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
Trap Responses	One per response	HPOV (COTS)	Managed Devices	SNMP query responses from managed objects. To promote security, DCE Remote Procedure Calls (RPCs) are used instead of SNMP to communicate with managed devices. The RPC is between the OpenView management station and the subagents on each host.
Updates	One per update	<i>Databases:</i> OV Object, Topology, Map	<i>Processes:</i> EcMsCmEcsd, EcMsCmOvmap, HPOV (COTS) <i>Libraries:</i> Ovw, Ovsnmp	The OV Object DB, Topology DB, and Map DB are updated via the HPOV API and the EcMsCmEcsd and EcMsCmOvmap processes update the OV Object DB only.
Topology Traps	One per topology state change	<i>Process:</i> EcMsCmEcsd <i>Library:</i> Ovw <i>Class:</i> MsCmMgrBkg	<i>Process:</i> Trapd (COTS)	The Trapd process forwards topology events in an SNMP trap format to the EcMsCmEcsd server. The Trapd process receives updated topology traps.
Updated Topology Traps	One per topology state change	<i>Process:</i> Trapd (COTS)	<i>Process:</i> EcMsCmEcsd <i>Libraries:</i> Ovsnmp, Ovw <i>Classes:</i> MsCmMgrBkg, MsCmTopoEvent	The EcMsCmEcsd server creates an updated topology trap once it ensures the object exists in the object database.

Table 4.9.1.6.1.5-2. Network and Enterprise Management Framework Process Interface Events (1 of 2)

Event	Event Frequency	Interface	Initiated By	Event Description
Send Event Information	One per log information	<i>Process:</i> Tivoli Event Server (COTS)	<i>Process:</i> Tivoli Log File Adapter (COTS)	The significant event (e.g., "server died," "connection lost," or "license due to expire mm/dd/yy) information saved by the Tivoli Log File Adapter is extracted by the Tivoli Event Server. Events can be logged to a text file if configured within the event definition/configuration via an Internal Tivoli API call.
Scan & Retrieve Log Information	Per selected log file	Managed Hosts	<i>Process:</i> Tivoli Log File Adapter (COTS)	The Tivoli Log File Adapter periodically checks COTS log files to determine if significant events (e.g., "server died," "connection lost," or "license due to expire mm/dd/yy) have occurred and saves these events.
Store Events	One per store events	<i>Data Store:</i> Tivoli Event DB	<i>Process:</i> Tivoli Event Server (COTS)	Events received by the Tivoli Event Server, via an Internal Tivoli API call, are processed and stored in the Event database.
Display Events	One per events	<i>Data Store:</i> Tivoli Event DB	<i>Process:</i> Tivoli Event Console (COTS)	The M&O staff uses the Tivoli Enterprise Console to retrieve events from the Event database and display them.
Retrieve Host Configuration	One per network host	<i>Data Store:</i> Tivoli Managed Region Configuration	<i>Process:</i> Tivoli Managed Region Server (COTS)	The Tivoli Managed Region (TMR) Server retrieves the host configuration from the Tivoli Management Region Configuration file via an Internal Tivoli API call.
Run Processes	One per run processes	Scripts	<i>Process:</i> Tivoli Managed Region Server (COTS)	The TMR server is able to execute scripts on managed hosts via an Internal Tivoli API call. An example of this would be to periodically remove core files.

Table 4.9.1.6.1.5-2. Network and Enterprise Management Framework Process Interface Events (2 of 2)

Event	Event Frequency	Interface	Initiated By	Event Description
Retrieve Performance Metrics Data	Per Operations Staff request	Managed Hosts	<i>Process:</i> Tivoli Sentry (COTS)	The Tivoli Sentry retrieves the performance metrics data (data describing host performance measurements) from the managed hosts.
Request Operational Mode	One per request of operational mode	<i>Data Stores:</i> Avail. Mode File, Active Mode File	<i>Process:</i> EcMsCmModeMgrG UI <i>Library:</i> MsAgDeputyGate <i>Class:</i> MsCmModeManager	The Mode management GUI updates the Active Mode and Available Mode files that are NFS mounted across all managed servers via the Rogue Wave file access API.
Send Threshold Exceeded Events	One per threshold event exceed	<i>Process:</i> Tivoli Event Server (COTS)	<i>Process:</i> Tivoli Sentry (COTS)	Events (data describing host performance measurements that exceed operator-defined levels) are sent to the Tivoli Event Server.

4.9.1.6.1.6 Network and Enterprise Management Framework Data Stores

Table 4.9.1.6.1.6-1 provides descriptions of the data stores used in the Network and Enterprise Management Framework architecture diagram.

Table 4.9.1.6.1.6-1. Network and Enterprise Management Framework Data Stores (1 of 2)

Data Store	Type	Functionality
Object DB	Database	The Object database contains all the objects (physical and logical) in the network that have been discovered by OpenView NNM.
Map DB	Database	The Map database stores presentation information for each object stored in the object database. A map is a collection of objects from the Object database along with their relationships. A map contains a subset of all the objects in the Object database.
Topology DB	Database	The Topology database contains an electronic representation of the topology of the infrastructure of the network. This includes all entities with IP addresses.
Tivoli Management Region Configuration	File	The Tivoli Management Region (TMR) Configuration defines the managed network host configuration and the TMR configuration is defined via an initialization procedure.

Table 4.9.1.6.1.6-1. Network and Enterprise Management Framework Data Stores (2 of 2)

Data Store	Type	Functionality
Tivoli Event DB	Database	The Tivoli Event DB contains those events forwarded by the Tivoli Event Server. These events can be retrieved and displayed by the M&O Staff for review on the TEC.
Avail. Mode File	File	Available Mode File contains the modes defined for use in the ECS and that can be changed through the Mode Management GUI.
Active Mode File	File	Active Mode File contains those modes that are activated.

4.9.1.6.2 MCI - Security Service Computer Software Component Description

4.9.1.6.2.1 Security Service Functional Overview

Security Service monitoring in the ECS is accomplished through several commercial and public domain programs. The programs vary from aiding in administration of DCE, assisting the user in choosing a password difficult to break, monitoring key system files for signs of tampering and probing hosts for well known security violations.

4.9.1.6.2.2 Security Service Context

Figure 4.9.1.6.2.2-1 is the Security Service context diagram. The diagram shows the events sent to the Security Service from the host operating system, communications devices, and the M & O staff and the events the Security Service sends to the M & O staff. Table 4.9.1.6.2.2-1 provides descriptions of the interface events shown in the Security Service context diagram.

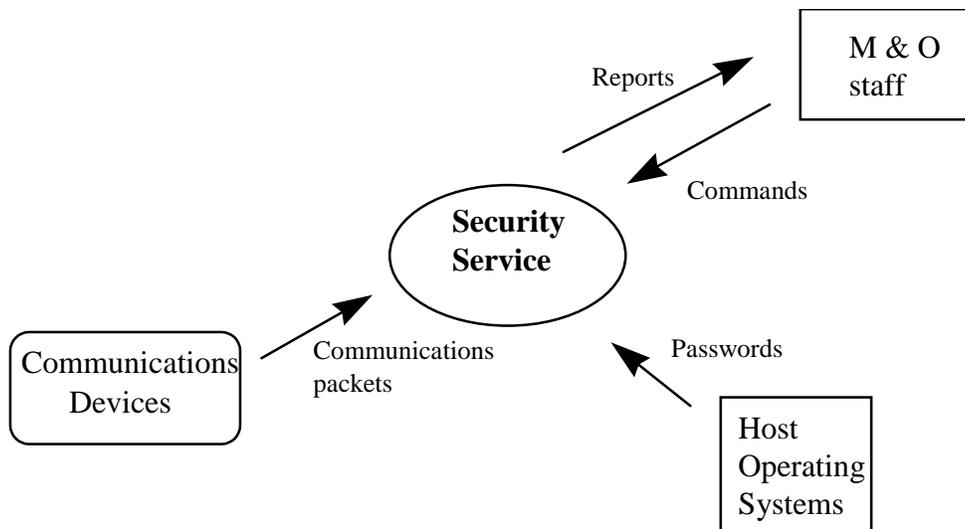


Figure 4.9.1.6.2.2-1. Security Service Context Diagram

Table 4.9.1.6.2.2-1. Security Service Interface Events

Event	Interface Event Description
Reports	The Security Service CSC utilities perform their functions and report results to the M&O Staff.
Commands	The M&O Staff issues commands to the Security Service CSC utilities to exercise system security setup.
Passwords	A password list is obtained from the Network Information Service (NIS) master by issuing a ypcat password command. This list is analyzed to see if decryption of a password is possible.
Communications packets	A packet reaches the ECS host from either an external source or from a host within the same site. The Security Service CSC analyzes packets for authorized sending sources.

4.9.1.6.2.3 Security Service Architecture

Figure 4.9.1.6.2.3-1 is the Security Service architecture diagram. The diagram shows the events sent to the Security Service processes and the events the Security Service processes send to other processes.

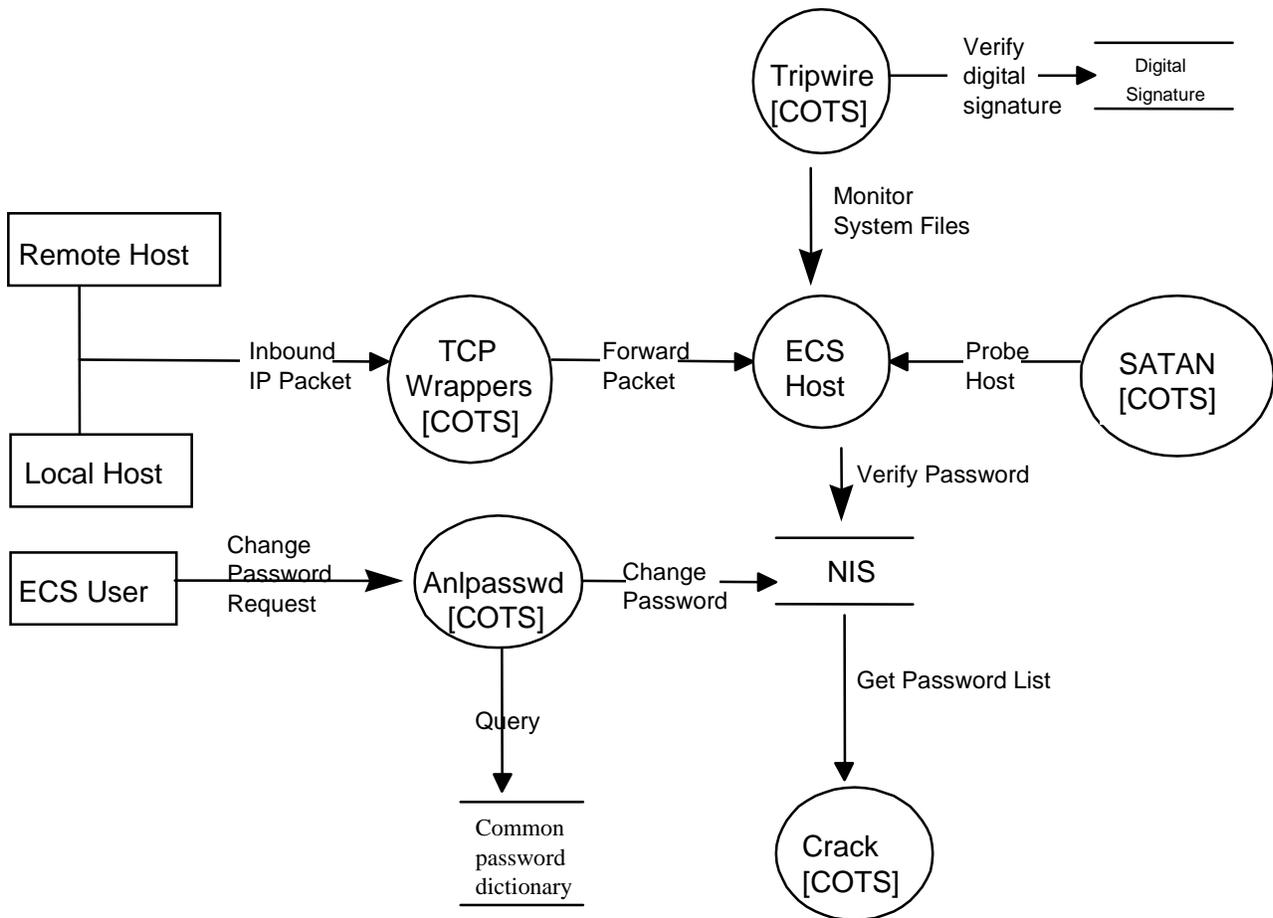


Figure 4.9.1.6.2.3-1. Security Service Architecture Diagram

4.9.1.6.2.4 Security Service Process Descriptions

Table 4.9.1.6.2.4-1 provides descriptions of the processes shown in the Security Service architecture diagram.

Table 4.9.1.6.2.4-1. Security Service Processes

Process	Type	COTS / Developed	Functionality
Anlpasswd	Other	COTS	Anlpasswd is a replacement for the standard UNIX passwd and ypasswd programs. Anlpasswd provides functionality by checking the selected user password to determine if the password is common or trivial and easy to break.
TCP Wrappers	Other	COTS	TCP Wrappers verifies the origin of incoming IP packets from an authorized host for services TCP Wrappers can filter. TCP Wrappers runs on each ECS host (UNIX) at a specific site.
Tripwire	Other	COTS	Tripwire periodically verifies that system files have not been altered. Tripwire is able to catch modifications by verifying the current and stored digital signatures of the command.
SATAN	GUI / Other	COTS	An M&O Staff member runs SATAN periodically to determine if any common vulnerability exists on ECS controlled hosts. The results are displayed in a web browser upon completion of a scan.
Crack	Other	COTS	An M&O Staff member runs Crack periodically to search for passwords that can be broken and were not caught by Anlpasswd.

4.9.1.6.2.5 Security Service Process Interface Descriptions

Table 4.9.1.6.2.5-1 provides descriptions of the interface events shown in the Security Service architecture diagram.

Table 4.9.1.6.2.5-1. Security Service Process Interface Events (1 of 2)

Event	Event Frequency	Interface	Initiated By	Event Description
Verify digital signature	Once per verify digital signature	<i>Data Store:</i> Digital Signature	<i>Process:</i> Tripwire (COTS)	The newly computed digital signature of a file is verified against the stored historical copy of the same file via a Tripwire internal call.
Monitor system files	One per monitor system files	ECS Host	<i>Process:</i> Tripwire (COTS)	Critical system files are watched periodically for changes in their digital signature that could signal a maliciously altered system file or service on an ECS Host.

Table 4.9.1.6.2.5-1. Security Service Process Interface Events (2 of 2)

Event	Event Frequency	Interface	Initiated By	Event Description
Probe host	One per probe host	ECS Host	<i>Process:</i> SATAN (COTS)	SATAN is used on an ad hoc basis, via a Netscape interface, to probe the hosts within a network to determine if any common security violations exist.
Forward Packet	One per forward packet	Inetd – UNIX daemon on an ECS Host	TCP Wrappers (COTS)	If the IP header indicates the packet originates from a host that has not been blocked by TCP Wrappers, the packet is forwarded via the appropriate internet service to an ECS Host.
Verify password	Once per verify password	<i>Data Store:</i> NIS	ECS Host	A request is sent from the ECS host, via a NIS system call, to the NIS database to verify that a login password is valid.
Get password list	One per get password list	<i>Data Store:</i> NIS	<i>Process:</i> Crack (COTS)	A password list is obtained from the NIS master by issuing a ypcat passwd command (NIS system call). This list is run through crack to see if crack is able to decrypt any user's password.
Change password	One per change password	<i>Data Store:</i> NIS	<i>Process:</i> Anlpasswd (COTS)	After the new password passes the Anlpasswd validation process, a request is sent to the NIS master, via a NIS system call, to modify the user's password.
Query	One per query	<i>Data Store:</i> Common Password Dictionary	<i>Process:</i> Anlpasswd (COTS)	The Anlpasswd makes a NIS system call to check the common word dictionary to ensure the attempted new password is not in this list.
Change password request	One per change password request	<i>Process:</i> Anlpasswd	User/Comm and line	An ECS user attempts to change their password and the request is verified by Anlpasswd that the new password does not contain any trivial or easy to guess password.
Inbound IP packet	One per inbound IP packet	<i>Process:</i> TCP Wrappers (COTS)	Remote or Local Host	A packet reaches the TCP Wrappers process from either an external source (remote host) or from a host within the same site via TCP/IP protocols.

4.9.1.6.2.6 Security Service Data Stores

Table 4.9.1.6.2.6-1 provides descriptions of the data stores shown in the Security Service architecture diagram.

Table 4.9.1.6.2.6-1. Security Service Data Stores

Data Store	Type	Functionality
NIS	Database	This UNIX service enables a common login on a number of machines and mapping for a user's Network File System (NFS) mounted home directory. The passwd map stores a user's login id, group id, and password in the NIS database.
Common password dictionary	Database	This sorted text file contains common words used by a user as a password. Anpasswd verifies that the new password change does not include a word listed in the Anpasswd file.
Digital Signature	Database	This proprietary database is used by Tripwire to record the digital signature for each system file it monitors.

4.9.1.6.3 MCI - Accountability Management Service Computer Software Component Description

4.9.1.6.3.1 Accountability Management Service Functional Overview

The Accountability Management Service supports User Registration and Order Tracking.

User Registration

ECS provides for two generic classes of users: guest users and registered users. Guest users are not formally registered. Registered users have submitted requests for a registered user account and have accounts, based on an approval process. Registered users can access services and products beyond those available to guest users.

Guest users can submit a request for a registered user account. The submitted request is captured at the SMC in a database of pending requests. The Operations staff accesses the database of pending requests and creates registered user accounts for approved requests.

The user registration server supports the creation, modification and maintenance of profiles for each registered user. The user profile is maintained at the SMC and replicated at each DAAC. Each DAAC is capable of browsing foreign user profiles locally, but only capable of modifying user profiles at the SMC for which it is the designated home DAAC.

The user registration GUI enables the DAAC Operations staff to view user registration requests (at the SMC only) and registered user profiles (at the DAAC and the SMC). The user profile information includes the user's name, identification code, primary DAAC, organizational affiliation, investigating group (such as an instrument team) affiliation (if any), assigned project, mailing address, shipping address for data or product order distribution media preferences for product orders, telephone number and electronic mail address (if any).

The Accountability Management Service enables the various subsystems to request user profile information such as the user's electronic mail address and the shipping address for product order or data distribution.

Order Tracking

The Order Tracking service provides the capability to track a product order's status during request processing. The Order Tracking service centralizes order status in the MSS database instead of going to each subsystem to collect it. The Order Tracking Server provides the public interface to other subsystems for updating order and request status in real time. The Order Tracking GUI enables order status browsing by user name, orderId, or the orderId's associated requestId.

Order Tracking has interfaces with other subsystems to provide access to order and request status. This tracking information is saved in the Order Tracking database.

4.9.1.6.3.2 Accountability Management Service Context

Figure 4.9.1.6.3.2-1 is the Accountability Management context diagram. The diagram shows the events sent to the Accountability Management and the events the Accountability Management sends to other CSCIs or CSCs. Table 4.9.1.6.3.2-1 provides descriptions of the interface events shown in the Accountability Management context diagram.

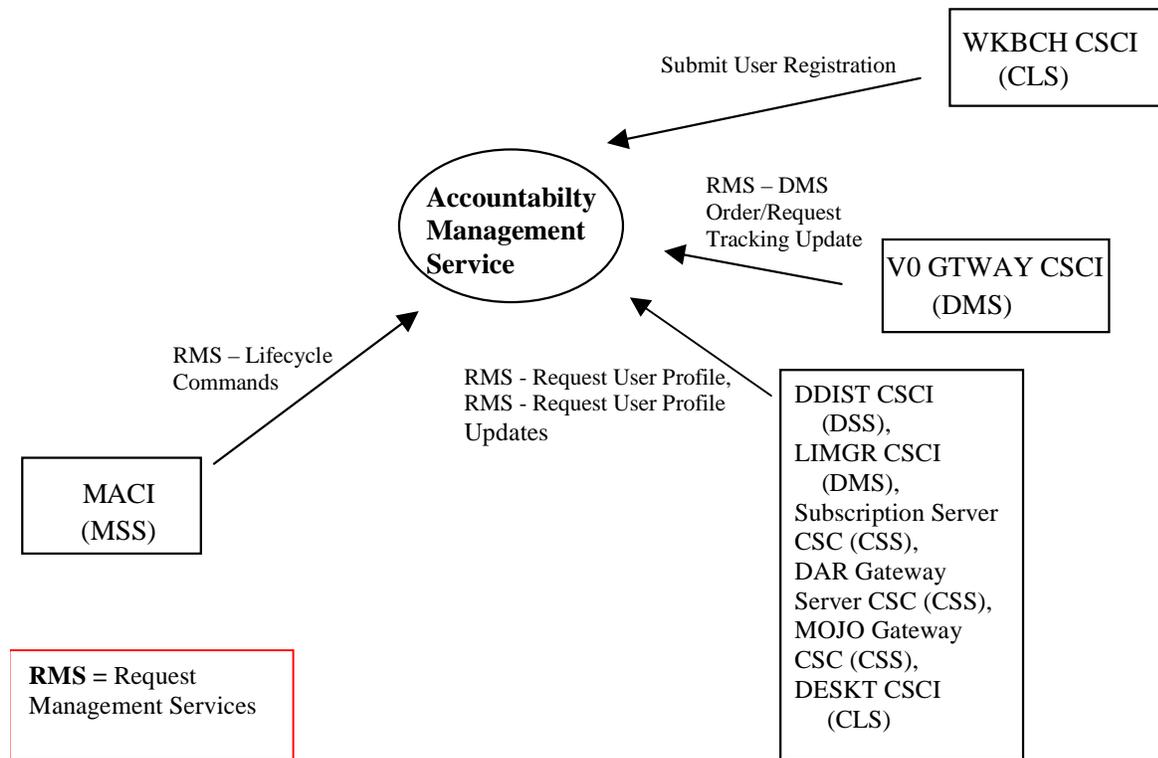


Figure 4.9.1.6.3.2-1. Accountability Management Service Context Diagram

Table 4.9.1.6.3.2-1. Accountability Management Service Interface Events

Event	Interface Event Description
Submit User Registration	Guest users submit ECS registration requests through the WKBCH CSCI.
Request Management Services	<p>The MACI and MCI provide a basic management library of services to the CSCIs/CSCs, implemented as client or server applications, using the DCCI CSCI Process Framework. The basic management library of services include:</p> <ul style="list-style-type: none"> • Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training). <p>The MCI also interfaces with other CSCIs/CSCs to perform the following:</p> <ul style="list-style-type: none"> • DMS Order/Request Tracking Update - The V0 GTWAY CSCI interfaces with Accountability Management Service Order/Request Tracking service to create a user product order. • User Profile Request - The Accountability Management Service provides requesting CSCIs/CSCs with User Profile parameters such as e-mail address and shipping address to support their processing activities. • User Profile Updates – The MCI receives user profile parameter updates from a user and makes the updates in the user profile database. This capability is available only at the SMC.

4.9.1.6.3.3 Accountability Management Service Architecture

Figure 4.9.1.6.3.3-1 is the Accountability Management Service architecture diagram. The diagram shows the events sent to the Accountability Management Service processes and the events the Accountability Management Service processes send to other processes.

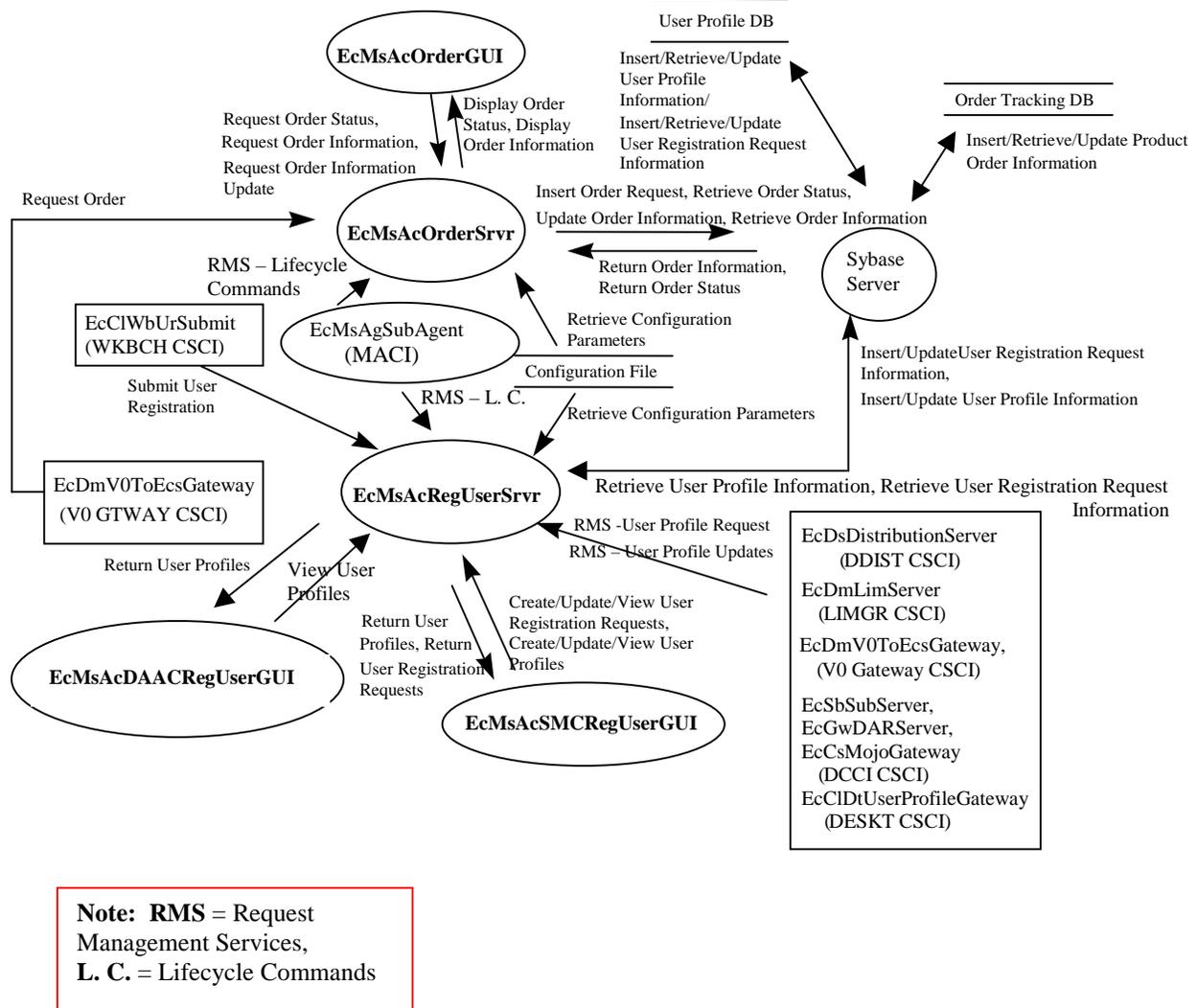


Figure 4.9.1.6.3.3-1. Accountability Management Service Architecture Diagram

4.9.1.6.3.4 Accountability Management Service Processes

Table 4.9.1.6.3.4-1 provides the descriptions of the processes shown in the Accountability Management Service architecture diagram.

Table 4.9.1.6.3.4-1. Accountability Management Service Processes (1 of 2)

Process	Type	COTS / Developed	Functionality
EcMsAcRegUserSrvr	Server	Developed	<p>The User Registration Server provides an internal interface to the User Registration GUI and an external interface to other CSCIs/CSCs. The functions are:</p> <ol style="list-style-type: none"> 1. Insert, delete, update, retrieve user registration request 2. Insert, delete, update, retrieve registered user profile 3. Retrieve a list of user registration requests 4. Retrieve a list of registered user profiles 5. Change V0 gateway password <p>The EcMsAcRegUserSrvr supports:</p> <ul style="list-style-type: none"> • Single requests at a time • Multiple concurrent requests • Asynchronous request processing • Request processing de-coupled from an RPC thread • Multiple threads within a single request
EcMsAcSMCRegUserGUI	GUI	Developed	<p>The User Registration graphical user interface enables the viewing and updating of user profiles. The GUI enables the user to :</p> <ol style="list-style-type: none"> 1. Add an ECS user and send e-mail notification 2. Delete an ECS user 3. Modify an ECS user profile 4. Change the V0 gateway password 5. Change ASTER category and send e-mail 6. Change the DAR privilege <p>The ASTER e-mail address is stored in the Accountability configuration file. The Accountability configuration file is read in when the Accountability GUI is started up.</p>
EcMsAcDAACRegUserGUI	GUI	Developed	<p>The User Registration graphical user interface enables the viewing of user profiles. The GUI enables the user to list and view ECS user profiles.</p>

Table 4.9.1.6.3.4-1. Accountability Management Service Processes (2 of 2)

Process	Type	COTS / Developed	Functionality
EcMsAcOrderSrvr	Server	Developed	<p>The Order Tracking Server provides an external interface to other CSCIs/CSCs. The functions are:</p> <ol style="list-style-type: none"> 1. Insert, delete, update, retrieve order 2. Insert, delete, update, retrieve request 3. Retrieve a list of orders 4. Retrieve a list of requests 5. Update order status 6. Update request status <p>The EcMsAcOrderSrvr supports:</p> <ul style="list-style-type: none"> • Single requests at a time • Multiple concurrent requests • Asynchronous request processing • Request processing de-coupled from an RPC thread • Multiple threads within a single request
EcMsAcOrderGUI	GUI	Developed	<p>This graphical user interface enables the user to retrieve the order and request from the Accountability database. The following functions are available:</p> <ol style="list-style-type: none"> 1. Retrieve order and request by order id or request id. 2. Retrieve order and request by user name. 3. Retrieval can be filtered by order type, order status and request status.
Sybase Server	Server	COTS	<p>The SQL server supporting access to the Sybase DBMS. The interface between processes and the databases for storage and retrieval of data or information.</p>

4.9.1.6.3.5 Accountability Management Service Process Interface Descriptions

Table 4.9.1.6.3.5-1 provides descriptions of the interface events shown in the Accountability Management Service architecture diagram.

**Table 4.9.1.6.3.5-1. Accountability Management Service Process Interface Events
(1 of 9)**

Event	Event Frequency	Interface	Initiated By	Event Description
Insert/Retrieve /Update User Profile Information	One per insert/retrieve/update of user profile	<i>Data Store:</i> User Profile DB	<i>Process:</i> Sybase Server (COTS)	The Sybase Server stores, retrieves, or updates user profile information by request.
Insert/Retrieve /Update User Registration Request Information	One per insert/retrieve/update of user registration request information	<i>Data Store:</i> User Profile DB	<i>Process:</i> Sybase Server (COTS)	The Sybase Server stores, retrieves, or updates information regarding user registration requests.
Insert/Retrieve /Update Product Order Information	One per order request	<i>Data Store:</i> Order Tracking DB	<i>Process:</i> Sybase Server (COTS)	The Sybase Server inserts, retrieves, and updates product order information in the Order Tracking DB.
Insert Order Request	One per insert order request	Sybase Server (COTS)	<i>Process:</i> EcMsAcOrderSrvr <i>Libraries:</i> MsAcCInt, MsAcComm	The EcMsAcOrderSrvr submits a request to the Sybase Server to insert a product order request into the Order tracking database (DB).
Retrieve Order Status	One per product order	Sybase Server (COTS)	<i>Process:</i> EcMsAcOrderSrvr <i>Libraries:</i> MsAcCInt, MsAcComm	The EcMsAcOrderSrvr sends a request to the Sybase Server to retrieve the status of an order placed by a user.
Update Order Information	One per update of order information	Sybase Server (COTS)	<i>Process:</i> EcMsAcOrderSrvr <i>Libraries:</i> MsAcCInt, MsAcComm	The EcMsAcOrderSrvr submits a request to the Sybase Server to update order information in the Order tracking database (DB).
Retrieve Order Information	One per update of order information	Sybase Server (COTS)	<i>Process:</i> EcMsAcOrderSrvr <i>Libraries:</i> MsAcCInt, MsAcComm	The EcMsAcOrderSrvr submits a request to the Sybase Server to retrieve order information from the Order tracking database (DB).

**Table 4.9.1.6.3.5-1. Accountability Management Service Process Interface Events
(2 of 9)**

Event	Event Frequency	Interface	Initiated By	Event Description
Return Order Information	One per return of order information	<i>Process:</i> EcMsAcOrderSrvr <i>Libraries:</i> MsAcCInt, MsAcComm	<i>Process:</i> Sybase Server (COTS)	The Sybase Server returns product order information per operations request to the EcMsAcOrderSrvr.
Return Order Status	One per return order status	<i>Process:</i> EcMsAcOrderSrvr <i>Libraries:</i> MsAcCInt, MsAcComm	<i>Process:</i> Sybase Server (COTS)	The Sybase Server returns product order status per operations request to the EcMsAcOrderSrvr.
Retrieve Configuration Parameters	One per retrieve of configuration parameters	<i>Data Store:</i> Configuration File	<i>Processes:</i> EcMsAcRegUserSrvr, EcMsAcOrderSrvr <i>Libraries:</i> MsAcCInt, MsAcComm	The EcMsAcRegUserSrvr and the EcMsAcOrderSrvr retrieve default parameters from the configuration file upon startup or the default parameters can be changed by request and the servers restarted.
Insert/Update User Registration Request Information	One per user registration request insert/update.	Sybase Server (COTS)	<i>Process:</i> EcMsAcRegUserSrvr <i>Libraries:</i> MsAcCInt, MsAcComm	The EcMsAcRegUserSrvr sends requests to the Sybase server to add or modify user registration request information in the User Profile database (DB).
Insert/Update User Profile Information	One per user insert/update profile	Sybase Server (COTS)	<i>Process:</i> EcMsAcRegUserSrvr <i>Libraries:</i> MsAcCInt, MsAcComm	The EcMsAcRegUserSrvr sends requests to the Sybase Server to add or modify user profile data in the User Profile database (DB).

**Table 4.9.1.6.3.5-1. Accountability Management Service Process Interface Events
(3 of 9)**

Event	Event Frequency	Interface	Initiated By	Event Description
Retrieve User Profile Information	One per profile request	Sybase Server (COTS)	<i>Process:</i> EcMsAcRegUserSrvr <i>Libraries:</i> MsAcClnt, MsAcComm	The EcMsAcRegUserSrvr receives from the Sybase Server the user profile information requested by the M&O Staff.
Retrieve User Registration Information	One registration request	Sybase Server (COTS)	<i>Process:</i> EcMsAcRegUserSrvr <i>Libraries:</i> MsAcClnt, MsAcComm	The EcMsAcRegUserSrvr receives from the Sybase Server the user registration information requested by the M&O Staff.
Request Management Services (RMS)				The EcMsAgSubAgent, EcMsAcOrderSrvr and EcMsAcRegUserSrvr provide a basic management library of services to the processes, implemented as client or server applications, using the DCCI CSCI Process Framework. The basic management library of services include:

**Table 4.9.1.6.3.5-1. Accountability Management Service Process Interface Events
(4 of 9)**

Event	Event Frequency	Interface	Initiated By	Event Description
(RMS – cont.)		<i>Processes:</i> EcMsAcOrderSrvr, EcMsAcRegU serSrvr,	<i>Process:</i> EcMsAgSubAgent <i>Library:</i> EcAgInstrm <i>Class:</i> EcAgManager	<ul style="list-style-type: none"> Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training).

**Table 4.9.1.6.3.5-1. Accountability Management Service Process Interface Events
(5 of 9)**

Event	Event Frequency	Interface	Initiated By	Event Description
(RMS – cont.)		<i>Process:</i> EcMsAcRegUserSrvr, <i>Libraries:</i> MsAcCInt, MsAcComm <i>Class:</i> EcAcProfileMgr	CSS Process: EcGwDARServer <i>Class:</i> EcGwDARGatewayRequest_S CSS Process: EcSbSubServer <i>Class:</i> EcSbSr DMS Process: EcDmV0ToEcsGateway <i>Class:</i> DmGwRequestReceiver DMS Process: EcDmLimServer <i>Library:</i> DmLmReqProc <i>Class:</i> DmLmProductPlan CLS Process: EcCIDtUserProfileGateway <i>Class:</i> CIDtUserProfileServer CSS Process: EcCsMojoGateway <i>Library:</i> EcCsMojoGateway <i>Class:</i> EcMjRetrieveProfileProxy DSS Process: EcDsDistributionServer <i>Library:</i> DsDdSSH <i>Class:</i> DsDdMedia	<ul style="list-style-type: none"> User Profile Request - The EcMsAcRegUserSrvr provides requesting processes with user profile parameters such as e-mail address and shipping address to support their processing activities.

**Table 4.9.1.6.3.5-1. Accountability Management Service Process Interface Events
(6 of 9)**

Event	Event Frequency	Interface	Initiated By	Event Description
(RMS – cont.)		<i>Process:</i> EcMsAcRegUserSrvr, <i>Libraries:</i> MsAcCInt, MsAcComm <i>Class:</i> EcAcProfileMgr	CSS Process: EcGwDARServer <i>Class:</i> EcGwDARGatewayRequest_S CSS Process: EcSbSubServer <i>Class:</i> EcSbSr DMS Process: EcDmV0ToEcsGateway <i>Class:</i> DmGwRequestReceiver DMS Process: EcDmLimServer <i>Library:</i> DmLmReqProc <i>Class:</i> DmLmProductPlan CLS Process: EcCIDtUserProfileGateway <i>Class:</i> CIDtUserProfileServer CSS Process: EcCsMojoGateway <i>Library:</i> EcCsMojoGateway <i>Class:</i> EcMjRetrieveProfileProxy DSS Process: EcDsDistributionServer <i>Library:</i> DsDdSSh <i>Class:</i> DsDdMedia	<ul style="list-style-type: none"> User Profile Updates - The EcMsAcRegUserSrvr provides requesting processes with updates to user profile parameters such as e-mail address and shipping address to support their processing activities.

**Table 4.9.1.6.3.5-1. Accountability Management Service Process Interface Events
(7 of 9)**

Event	Event Frequency	Interface	Initiated By	Event Description
Create/Update/View User Registration Requests	One per create/update/view user registration request	<i>Process:</i> EcMsAcRegUserSrvr <i>Libraries:</i> MsAcCInt, MsAcComm	M&O staff <i>Process:</i> EcMsAcSMCRegUserGUI <i>Libraries:</i> MsAcCInt, MsAcComm <i>Class:</i> MsAcRegUserUI	The M&O staff send requests, via the EcMsAcSMCRegUserGUI, to create, modify, or view user registration request information to the EcMsAcRegUserSrvr.
Create/Update/View User Profiles	One per create/update/view user profile	<i>Process:</i> EcMsAcRegUserSrvr <i>Libraries:</i> MsAcCInt, MsAcComm	M&O staff <i>Process:</i> EcMsAcSMCRegUserGUI <i>Libraries:</i> MsAcCInt, MsAcComm <i>Class:</i> MsAcRegUserUI	The M&O staff send requests to create, modify, or view user profile information via the EcMsAcSMCRegUserGUI.
Return User Profiles	One per return of user profile	<i>Processes:</i> EcMsAcSMCRegUserGUI, EcMsAcDAACRegUserGUI <i>Libraries:</i> MsAcCInt, MsAcComm	<i>Process:</i> EcMsAcRegUserSrvr <i>Libraries:</i> MsAcCInt, MsAcComm	The EcMsAcRegUserSrvr returns user profile information provided from the Sybase Server to the requester at the EcMsAcSMCRegUserGUI or EcMsAcDAACRegUserGUI.
Return User Registration Requests	One per return of user registration request	<i>Processes:</i> EcMsAcSMCRegUserGUI, EcMsAcDAACRegUserGUI <i>Libraries:</i> MsAcCInt, MsAcComm <i>Class:</i> MsAcRegUserUI	<i>Process:</i> EcMsAcRegUserSrvr <i>Libraries:</i> MsAcCInt, MsAcComm	The EcMsAcRegUserSrvr returns user registration request information provided from the Sybase Server to the requester at the EcMsAcSMCRegUserGUI.

**Table 4.9.1.6.3.5-1. Accountability Management Service Process Interface Events
(8 of 9)**

Event	Event Frequency	Interface	Initiated By	Event Description
Create/Delete DCE Accounts	One per DCE registration	DCE Security Server (COTS)	M&O Staff <i>Process:</i> EcMsAcRegUserGUI <i>Libraries:</i> MsAcCInt, MsAcComm <i>Class:</i> MsAcRegUserUI	The EcMsAcRegUserSvr sends requests to the DCE Security Server to create a DCE user account or delete a DCE user account.
View User Profiles	One per view user profile request	<i>Process:</i> EcMsAcRegUserSvr <i>Libraries:</i> MsAcCInt, MsAcComm	M&O staff <i>Process:</i> EcMsAcDAACRegUserSvr <i>Libraries:</i> MsAcCInt, MsAcComm	The M&O staff request, via the EcMsAcDAACRegUserGUI, to retrieve a user profile from the EcMsAcRegUserSvr.
Request Order	One per Request order	<i>Process:</i> EcMsAcOrderSvr <i>Libraries:</i> MsAcCInt, MsAcComm	<i>Process:</i> EcDmV0ToEcsGateway <i>Library:</i> RequestProcessing <i>Class:</i> DmGwAcquireRequest	The EcDmV0ToECSGateway sends a request for a product order to the EcMsAcOrderSvr.
Submit User Registration	One per registration request to MSS	<i>Process:</i> EcMsAcRegUserSvr <i>Libraries:</i> MsAcCInt, MsAcComm <i>Classes:</i> EcAcUsrRequestMgr, MsAcUsrRequest	<i>Script:</i> EcCIWbUrSubmit	A guest user requests to become a registered ECS user. The guest user must invoke the EcCIWbUr CGI script to enter registration information, invoke the EcCIWbUrConfirm CGI script to have the information confirmed, and invoke the EcCIWbUrSubmit CGI script to send the registration information to the EcMsAcRegUserSvr. The URT CGI script takes the confirmed user inputs and submits the registration request to the EcMsAcRegUserSvr.

**Table 4.9.1.6.3.5-1. Accountability Management Service Process Interface Events
(9 of 9)**

Event	Event Frequency	Interface	Initiated By	Event Description
Request Order Status	One per order request	<i>Process:</i> EcMsAcOrderSrvr <i>Libraries:</i> MsAcClnt, MsAcComm	<i>Process:</i> EcMsAcOrderGUI <i>Libraries:</i> MsAcClnt, MsAcComm	The EcMsAcOrderGUI sends requests for order status to the EcMsAcOrderSrvr.
Request Order Information	One per order request	<i>Process:</i> EcMsAcOrderSrvr <i>Libraries:</i> MsAcClnt, MsAcComm	<i>Process:</i> EcMsAcOrderGUI <i>Libraries:</i> MsAcClnt, MsAcComm	The EcMsAcOrderGUI sends requests for order information to the EcMsAcOrderSrvr.
Request Order Information Update	One per order request	<i>Process:</i> EcMsAcOrderSrvr <i>Libraries:</i> MsAcClnt, MsAcComm	<i>Process:</i> EcMsAcOrderGUI <i>Libraries:</i> MsAcClnt, MsAcComm	The EcMsAcOrderGUI sends requests for order information updates to the EcMsAcOrderSrvr.
Display Order Status	One per order request	<i>Process:</i> EcMsAcOrderGUI <i>Libraries:</i> MsAcClnt, MsAcComm	<i>Process:</i> EcMsAcOrderSrvr <i>Libraries:</i> MsAcClnt, MsAcComm	The EcMsAcOrderSrvr provides the order status information to the EcMsAcOrderGUI for display by the Operations Staff.
Display Order Information	One per order request	<i>Process:</i> EcMsAcOrderGUI <i>Libraries:</i> MsAcClnt, MsAcComm	<i>Process:</i> EcMsAcOrderSrvr <i>Libraries:</i> MsAcClnt, MsAcComm	The EcMsAcOrderSrvr provides the order information to the EcMsAcOrderGUI for display by the Operations Staff.

4.9.1.6.3.6 Accountability Management Service Data Stores

Table 4.9.1.6.3.6-1 provides descriptions of the data stores shown in the Accountability Management Service architecture diagram.

Table 4.9.1.6.3.6-1. Accountability Management Service Data Stores

Data Store	Type	Description
User Profile DB	Database	The User Profile DB contains requests for user registrations and it also contains the profile information including mailing addresses, e-mail address, and project affiliations of approved registered users.
Order Tracking DB	Database	The Order Tracking DB contains product orders and user requests with the associated current processing status.
Configuration File	Other	The Accountability software obtains configuration parameters from a configuration file at startup. It contains: <ol style="list-style-type: none"> 1. Host Name 2. Database login information 3. Application log level 4. Application log size 5. HP OpenView start up scripts 6. ASTER e-mail address

4.9.1.6.4 MCI - Trouble Ticket Computer Software Component Description

4.9.1.6.4.1 Trouble Ticket Functional Overview

Remedy’s Action Request System (ARS), commonly referred to as Remedy, implements the Trouble Ticketing service in the ECS. The GUI provided with Remedy enables the Operations staff to enter and track trouble tickets affecting both local and ECS system-wide resources. In addition, a custom web-based interface using the Remedy API enables ECS registered users to submit new trouble tickets and to obtain the current resolution status of their open trouble tickets. The delivered configuration of Remedy includes trouble escalation policies, operator notifications, and status reports to aid in the problem resolution process.

4.9.1.6.4.2 Trouble Ticket Context

Figure 4.9.1.6.4.2-1 is the Trouble Ticket (TT) context diagram. The ARS receives new trouble tickets from users. In addition, new trouble tickets are created using existing information from trouble tickets forwarded by other DAACs or an external system such as ASTER GDS, NSI, or Landsat 7. Remedy’s ARS mail daemon receives the e-mailed trouble tickets and submits them to the appropriate database. The ARS stores information in several Sybase tables – one table per schema used by the ARS. Notifications are automatically sent to the appropriate administrators upon creation and closure. An alarm notification is also sent if a trouble ticket has not been assigned to an investigator within a predetermined time period determined by ECS policy and procedures. The user who submits a Trouble Ticket is automatically notified upon creation of a trouble ticket and upon closure of the trouble ticket. A copy of a selected “closed” trouble ticket is forwarded to the SMC for consolidation with others in a closed trouble ticket database. The closed trouble ticket database serves as a knowledge base of past system problems and their resolutions. **Note:** For more detail on converting Trouble Tickets from Remedy to DDTS, please refer to the Work Instruction MO-1-003-6, section 6.3.5. This work instruction is located on the

EDHS Internal Server under the Project Instructions, work instructions (under the ECS Directive System), and Maintenance and Operations.

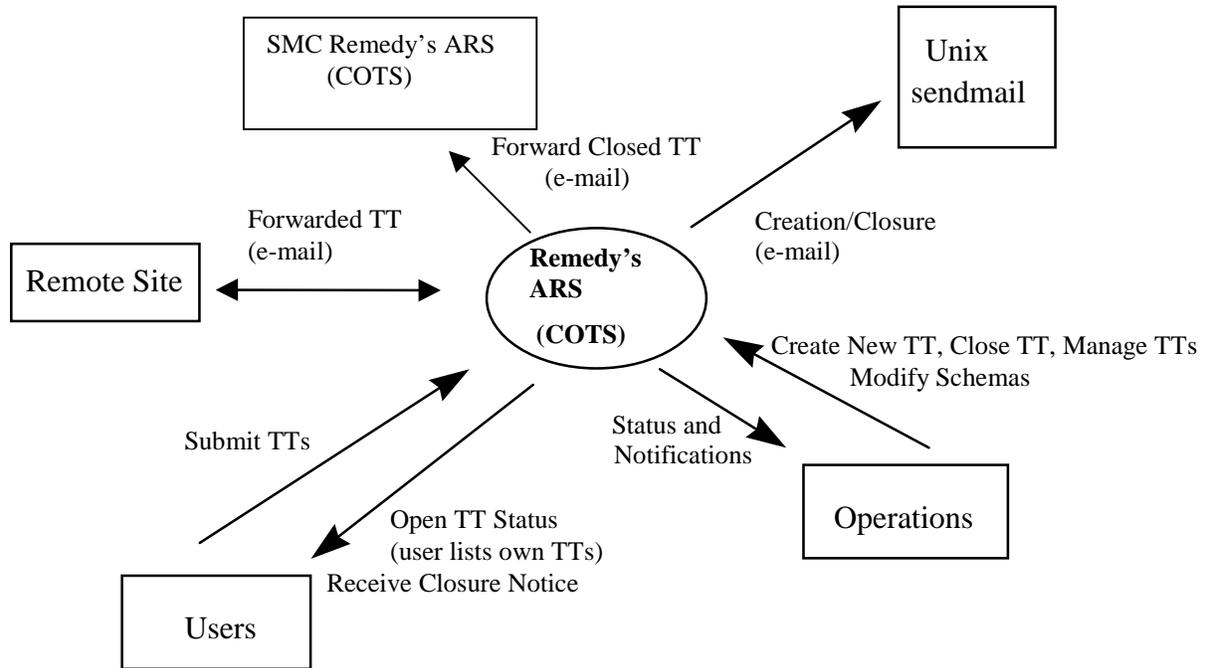


Figure 4.9.1.6.4.2-1. Trouble Ticket Context Diagram

Table 4.9.1.6.4.2-1 provides descriptions of the interface events shown in the Trouble Ticket context diagram.

Table 4.9.1.6.4.2-1. Trouble Ticket Interface Events (1 of 2)

Event	Interface Event Description
Creation/closure (e-mail)	E-mail is sent to the M&O staff member assigned to handle the TT upon creation or closure of the TT.
Create New TT	The M&O staff are able to submit new TTs using the aruser GUI supplied with the COTS software package.
Close TT	Upon resolution of a Trouble Ticket, the M&O staff member annotates the corrective actions in the TT schema and moves the TT to a Closed state. This triggers a Receive Closure Notice action.
Manage TTs	The TT administrator assigns open TTs to the appropriate M&O staff member. The M&O staff member receives notification by e-mail or the notifier tool based on preferences set in the Remedy User schema for that administrator.

Table 4.9.1.6.4.2-1. Trouble Ticket Interface Events (2 of 2)

Event	Interface Event Description
Modify Schemas	The TT administrator modifies schemas and screen layouts. This is not encouraged as it can produce incompatible TTs with other site schemas. Also, the TT administrator, to determine what escalations can be altered uses trouble ticket priority escalations and filters.
Status and Notifications	The M&O Staff and the Trouble Ticket (TT) Administrator can query all information concerning a particular TT through Remedy's aruser GUI. Notifications are sent to the staff member or group responsible for a particular stage of a Trouble Ticket. These notifications can include warnings that a TT has been assigned to a staff member or that a TT has been left in a particular state for too long. Notifications can be sent either by e-mail or the Remedy notifier GUI.
Open TT status (user lists own TTs)	An ECS user can query the TT database and find the current status of opened tickets.
Receive Closure Notice	An e-mail message is sent to the originator after the TT has been closed. This message includes the TT ID number and corrective actions taken.
Submit TTs	An ECS user can submit a TT via the custom web interface. This interface enables interaction with the User Profile Server. This is an alternative to calling the DAAC directly.
Forwarded TT (e-mail)	Using the mail template of the Remedy mail daemon (armaild), sites can create and forward new TTs to another site. This new TT has the original ID stored as a Unique Identifier.
Forward Closed TT (e-mail)	Using the Forward Closed TT to SMC capability, the site administrator can select and forward a copy of a closed TT to the SMC. The original TT's ID is stored as the ticket's unique identifier.

4.9.1.6.4.3 Trouble Ticket Architecture

Figure 4.9.1.6.4.3-1 is the Trouble Ticket architecture diagram. The diagram shows the events sent to the Remedy Action Request System (ARS) COTS process and the events the Remedy ARS COTS process sends to other processes (Remedy GUIs, daemons, and the Sybase Server).

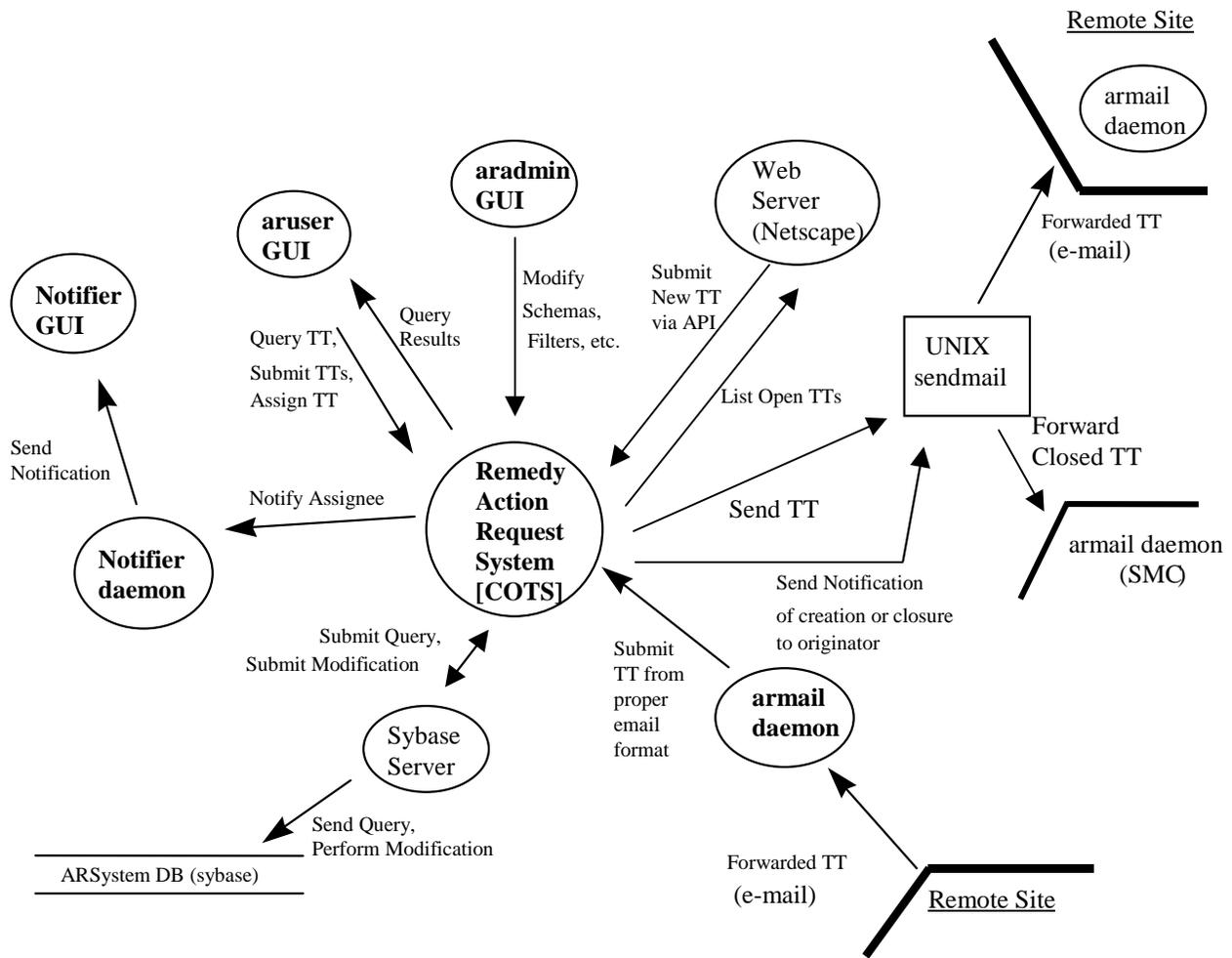


Figure 4.9.1.6.4.3-1. Trouble Ticket Architecture Diagram

4.9.1.6.4.4 Trouble Ticket Process Descriptions

Table 4.9.1.6.4.4-1 provides descriptions of the processes shown in the Trouble Ticket architecture diagram.

Table 4.9.1.6.4.4-1. Trouble Ticket Processes

Process	Type	COTS / Developed	Functionality
aruser GUI	GUI	COTS	The aruser GUI enables the M&O staff member to: <ol style="list-style-type: none"> 1. Submit a new TT 2. Query information about an existing TT 3. Move a TT to a Closed state and annotate the resolution The TT admin. uses this GUI to: <ol style="list-style-type: none"> 1. Add / Modify administrators in Remedy's User schema 2. Assign TTs to M&O staff
aradmin GUI	GUI	COTS	The TT administrator uses this GUI to: <ol style="list-style-type: none"> 1. Update schemas and aruser screen layouts 2. Update escalation policies and filters
Web Server (Netscape)	Server	Developed	This interface enables the ECS user access to the Trouble Ticket process without directly contacting an M&O staff member. The user is able to: <ol style="list-style-type: none"> 1. Submit a new TT 2. Query the status of an existing TT they submitted
Notifier GUI	GUI	COTS	This tool provides notification upon submission of a new TT or when an M&O staff member is assigned responsibility for a TT.
Notifier daemon	Server	COTS	The notifier daemon sends notifications to an M&O staff member's notifier GUI or by e-mail if the staff member's Remedy notification preference is set to e-mail.
Remedy ARS	Server	COTS	The Remedy ARS interacts with its associated GUIs via the provided Remedy daemons and the ARSystem DB. Error messages are logged to an aerror.log log file.
armail daemon	Server	COTS	The armail daemon monitors a mailbox (/var/spool/mail/arsystem) for incoming TTs formatted in the proper Remedy layout for TTs. Upon reception of a valid, formatted message, a new TT is created.
UNIX sendmail	Server	COTS	The sendmail daemon is an integral part of Remedy and must be properly configured for both e-mail notifications and TT forwarding to be accomplished.
Sybase Server	Server	COTS	Sybase accepts queries and requests for modifications to data in persistent storage in the ARSystem DB from the Remedy ARS.

4.9.1.6.4.5 Trouble Ticket Process Interface Descriptions

Table 4.9.1.6.4.5-1 provides descriptions of the interface events shown in the Trouble Ticket architecture diagram.

Table 4.9.1.6.4.5-1. Trouble Ticket Process Interface Events (1 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
Modify Schemas, Filters, etc.	One per modification to schemas, filters, etc.	<i>COTS:</i> Remedy ARS <i>Program:</i> MsTtManager	<i>Process:</i> aradmin GUI (COTS)	The TT administrator modifies schemas and screen layouts; escalation policies and filters via the Remedy supplied aradmin GUI.
Submit New TT via API	One per new TT submit via API	<i>COTS:</i> Remedy ARS <i>Programs:</i> MsTtHTMLItems, MsTtServiceRequestor, MsTtManager <i>Classes:</i> MsAcUsrProfile MsAcUsrProfileMgr_1_0	User Web Server (Netscape) - COTS	Alternatively, an ECS user can submit a TT via the custom web interface. This generates a new TT to begin the TT resolution process with a notification to the TT administrator of the new TT.
List open TTs	One per list open TTs	Web Server (Netscape) - COTS	User <i>COTS:</i> Remedy ARS <i>Programs:</i> MsTtHTMLMenu, MsTtHTMLItems, MsTtManager <i>Classes:</i> MsAcUsrProfile MsAcUsrProfileMgr_1_0	The ECS user can also query the Remedy system through the custom web interface to obtain a list of active TTs that they have submitted and their current status.
Forwarded TT (e-mail)	One per trouble ticket forwarded	<i>Process:</i> armail daemon (COTS)	<i>Process:</i> Unix Sendmail (COTS)	The M&O staff member sends the TT to another site to be archived or for escalation to a review board for action.
Forward Closed TT	One per Closed TT sent	<i>Process:</i> armail daemon (COTS)	Unix Sendmail (COTS)	The Site administrator sends a copy of a closed TT to the SMC.
Send TT	One per TT sent	<i>Process:</i> Unix Sendmail (COTS)	<i>COTS:</i> Remedy ARS <i>Program:</i> MsTtManager	Remedy ARS uses e-mail to send an open or closed TT to the appropriate site.

Table 4.9.1.6.4.5-1. Trouble Ticket Process Interface Events (2 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
Send Notification of creation or closure to originator	One per notification sent to originator	Unix Sendmail (COTS)	<i>COTS:</i> Remedy ARS <i>Program:</i> MsTtManager	The originators receive notification of creation or closure on their TT via e-mail sent using UNIX sendmail on the host where Remedy is running.
Submit TT from proper e-mail format	One per submit of TT in proper e-mail format	<i>COTS:</i> Remedy ARS <i>Program:</i> MsTtManager	<i>Process:</i> armail daemon (COTS)	The armail daemon monitors a mailbox (ARSystem) for new mail messages that conform to the TT mail exchange format. Upon receiving a valid message, a new TT is created that begins the TT resolution process with the TT administrator being notified of a new TT.
Submit Query	One per submit query	Sybase Server (COTS)	<i>COTS:</i> Remedy ARS <i>Program:</i> MsTtManager	After resolving the TT issue, the M&O Staff uses Remedy ARS to query the Sybase Server for the appropriate TT and receives the information from Sybase.
Submit Modification	One per modification	Sybase Server (COTS)	<i>COTS:</i> Remedy ARS <i>Program:</i> MsTtManager	The M&O Staff modifies the status (changes the status to Closed) of the appropriate TT.
Send Query	One per query	<i>Data Store:</i> ARSystem DB	Sybase Server (COTS)	The Sybase Server is used to query data in persistent storage.
Perform Modification	One per update	<i>Data Store:</i> ARSystem DB	Sybase Server (COTS)	The Sybase Server is used to modify data in persistent storage.
Notify Assignee	One per notify assignee	<i>Process:</i> Notifier daemon (COTS)	<i>COTS:</i> Remedy ARS <i>Program:</i> MsTtManager	A request is sent to the Remedy notifier daemon to notify the M&O staff member of responsibility for the TT.
Send Notification	One per send notification	<i>Process:</i> Notifier GUI (COTS)	<i>Process:</i> Notifier daemon (COTS)	The notifier daemon notifies the responsible M&O staff member via the notification GUI or by e-mail depending on User schema settings.

Table 4.9.1.6.4.5-1. Trouble Ticket Process Interface Events (3 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
Query TT	One per TT	<i>COTS:</i> Remedy ARS <i>Program:</i> MsTtManager	<i>Process:</i> aruser GUI (COTS)	The M&O staff member, upon receiving notification of a new TT, queries the Remedy TT schema to find the detailed information and process the TT.
Submit TTs	One per submit TT	<i>Program:</i> MsTtManager (COTS, Remedy ARS)	M&O staff <i>Process:</i> aruser GUI (COTS)	A new Trouble Ticket is created by M&O staff and entered into the Remedy system. A notification is sent to the TT administrator of the existence of a new TT.
Assign TT	One per assign TT	<i>Program:</i> MsTtManager (COTS, Remedy ARS)	M&O staff <i>COTS:</i> aruser GUI (COTS)	After receiving the notification, the TT administrator assigns the TT to an M&O staff member.
Query Results	One per query	<i>Process:</i> aruser GUI (COTS)	<i>COTS:</i> Remedy ARS <i>Program:</i> MsTtManager	TT query results are returned to the M&O staff via the Remedy ARS.

4.9.1.6.4.6 Trouble Ticket Data Stores

Table 4.9.1.6.4.6-1 provides descriptions of the data stores shown in the Trouble Ticket architecture diagram. Also, descriptions are provided for the configuration files used by the Trouble Ticket CSC.

Table 4.9.1.6.4.6-1. Trouble Ticket Data Stores

Data Store	Type	Functionality
ARSystem DB (Sybase)	Database	This database is controlled by Remedy and stores the information from each schema in its own table. There is no clear mapping of schema to table. The Sybase table names are usually similar to T1, T2, T13, etc. Information includes: <ol style="list-style-type: none"> 1. Trouble Ticket detailed information 2. Contact Log detailed information 3. User information for Remedy users 4. Group information for roles within Remedy 5. Menus used by the GUIs

4.9.1.6.5 MCI - Network Backup/Restore Computer Software Component Description

4.9.1.6.5.1 Network Backup/Restore Functional Overview

The Legato vendor's Networker package provides a suite of integrated tools for backup and recovery, archival and retrieval, and hierarchical storage management. The product supports multi-platform networks, contains a motif-based GUI with on-line help, and supports concurrent device support for parallel backup and recovery using up to 16 storage devices. Authorized users can perform scheduled and ad-hoc backups, recoveries, and other data management services. Networker software consists of two parts: a client portion, which runs on the systems to be backed up, and a server portion, which is the system to which the backup devices are connected. The client portion sends the data to be backed up to the server portion, which writes the data out to disk.

4.9.1.6.5.2 Network Backup/Restore Context

A context diagram is not applicable to the Network Backup/Restore CSC.

4.9.1.6.5.3 Network Backup/Restore Architecture

Figure 4.9.1.6.5.3-1 is the Network Backup/Restore architecture diagram. The diagram shows the events sent to the Network Server of the Network Backup/Restore CSC and the events the Network Server of the Network Backup/Restore CSC sends to other processes (network clients).

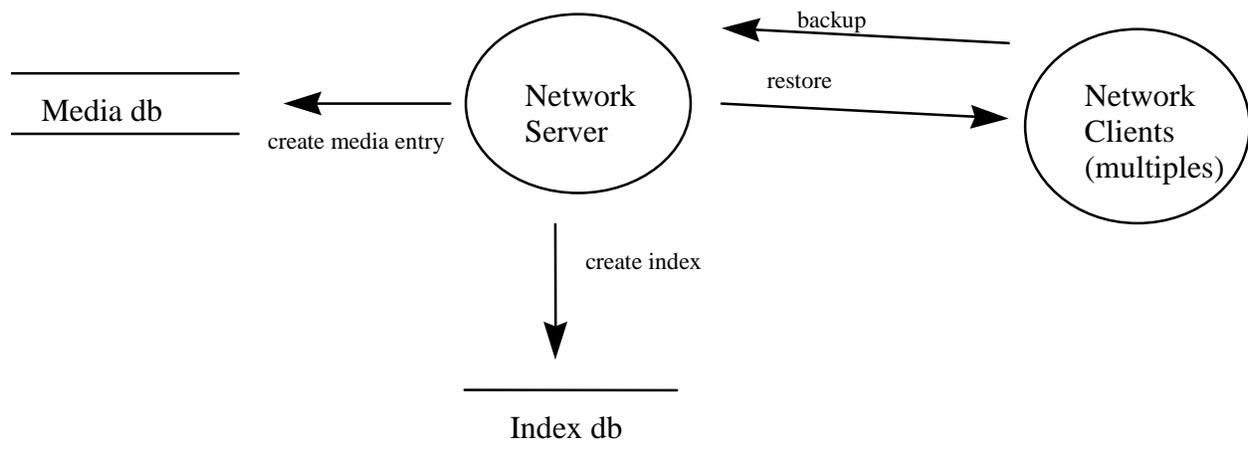


Figure 4.9.1.6.5.3-1. Network Backup/Restore Architecture Diagram

4.9.1.6.5.4 Network Backup/Restore Process Descriptions

Table 4.9.1.6.5.4-1 provides descriptions of the processes shown in the Network Backup/Restore architecture diagram.

Table 4.9.1.6.5.4-1. Network Backup/Restore Processes

Process	Type	COTS / Developed	Functionality
Network Server	Server	COTS	The server can support multiple requester backups simultaneously. An index file is created to enable the backup operator to quickly find the proper tape from which to restore files or file systems.
Network Clients	Client	COTS	On each host that is backed up by Network, a client portion is installed. The client portion can compress data before sending it to the server; however, doing so increases CPU usage on the client machine.

4.9.1.6.5.5 Network Backup/Restore Process Interface Descriptions

Table 4.9.1.6.5.5-1 provides descriptions of the interface events shown in the Network Backup/Restore architecture diagram.

Table 4.9.1.6.5.5-1. Network Backup/Restore Process Interface Events

Event	Event Frequency	Interface	Initiated By	Event Description
Backup	One per backup	<i>Process:</i> Network Server (COTS)	<i>Process:</i> Network Clients	Data is passed from the client to the server and archived to tape.
Restore	One per restore	<i>Process:</i> Network Clients	<i>Process:</i> Network Server (COTS)	Data is passed to the client to restore lost data from the tape backups.
Create Index	One per create index	COTS DB (Media db)	<i>Process:</i> Network Server (COTS)	While saving data to tape, an index is created that gives the tape identification for any version of a file that needs to be restored.
Create media entry	One per create media entry	COTS DB (Index db)	<i>Process:</i> Network Server	After saving data files (save sets) to tape, the Networker Server makes an entry in the media db identifying what save sets are on the tape.

4.9.1.6.5.6 Network Backup/Restore Data Stores

Table 4.9.1.6.5.6-1 provides descriptions of the data stores shown in the Network Backup/Restore architecture diagram.

Table 4.9.1.6.5.6-1. Network Backup/Restore Data Stores

Data Store	Type	Functionality
Index db	Other	This proprietary index enables the backup operator to determine the location of the file(s) needing to be restored without searching all the tapes in the stacker. This index includes version number information where appropriate.
Media db	Other	This media db tracks what file systems (save sets) are on each tape.

4.9.1.6.6 MCI - ASTER E-mail Header Handler Computer Software Component Description

4.9.1.6.6.1 ASTER E-mail Header Handler Functional Overview

As specified in the Interface Between the ECS Communications and Systems Management Segment (CSMS) and the ASTER GDS CSMS Ground System Management Subsystem (GSMS) ICD (209-CD-002-005, page 8-1), a formatted header is added to all e-mail exchanges between the ASTER GDS and the ECS sites. The header contains information on the send date and time, the sender and receiver ID, and a unique output message sequence number. The header is detailed in 209-CD-002-005, page 8-6. Although the header is a necessary part of the ASTER to ECS e-mail transfer protocol, it does not contain information needed by ECS sending or receiving applications. The header therefore is automatically added to ECS e-mail sent to ASTER and deleted from e-mail messages received from the ASTER GDS through the MSS provided ASTER e-mail header handler.

Using the ASTER to ECS e-mail transfer protocol, if a sequence number is skipped, the receiving site knows that a message has been lost and can request a re-transmission. A log of messages sent and received through this header process is maintained by the ECS. The copies of messages are maintained in a format that enables the M&O staff to re-send a requested transmission using a standard UNIX mail tool such as Netscape.

Addition and deletion of the ASTER standard e-mail header is accomplished by creating aliases used by the Unix sendmail daemon. For instance:

- A Trouble Ticket is to be sent to the ASTER GDS
- The Trouble Ticket is mailed to ECSTroubleTicket@<edc.gov>
- The sendmail daemon at <edc.gov> realizes that ECSTroubleTicket is an alias and filters the message through the AsterFilter.pl script
- The script adds the header information, logs and archives the message and forwards the message with header to the e-mail address specified in the alias, for example TroubleTicket@<aster.jp>

A similar flow exists for the removal of the header when receiving e-mail from the ASTER GDS. The System Management Center (SMC) at GSFC monitors and coordinates activities involving multiple sites and performs designated common support functions (e.g., receiving and sending mail from/to the ASTER GDS).

4.9.1.6.6.2 ASTER E-mail Header Handler Context

Figure 4.9.1.6.6.2-1 is the ASTER E-mail Header Handler context diagram. The diagram shows the events sent to the ASTER E-mail Header Handler and the events the ASTER E-mail Header Handler sends to ECS applications or the ASTER GDS. Table 4.9.1.6.6.2-1 provides descriptions of the interface events in the ASTER E-mail Header Handler context diagram.

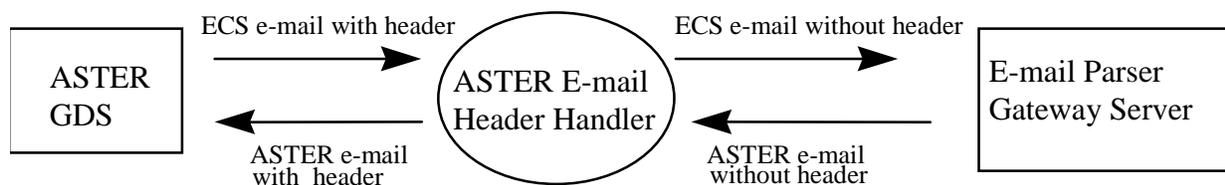


Figure 4.9.1.6.6.2-1. ASTER E-mail Header Handler Context Diagram

Table 4.9.1.6.6.2-1. ASTER E-mail Header Handler Interface Events

Interface	Interface Event Description
ECS e-mail without header	The header is removed from the inbound message, logged, and forwarded to the predefined ECS recipient of the e-mail alias.
ASTER e-mail without header	The E-Mail Parser Gateway Server to a predefined ASTER e-mail alias within the ECS, without a header, sends an e-mail message (e.g., Expedited Data Set Notification or EDN).
ASTER e-mail with header	The header is added to the e-mail message by the e-mail handler and forwarded to the ASTER destination.
ECS e-mail with header	An e-mail message (e.g., Expedited Data Set Request or EDR), containing an ASTER standard header, is sent from the ASTER GDS to a predefined e-mail alias at the ECS.

4.9.1.6.6.3 ASTER E-mail Header Handler Architecture

Figure 4.9.1.6.6.3-1 is the ASTER E-mail Header Handler architecture diagram. The diagram shows the events sent to the ASTER E-mail Header Handler processes and the events the ASTER E-mail Header Handler processes send to the System Management Center and the ASTER GDS.

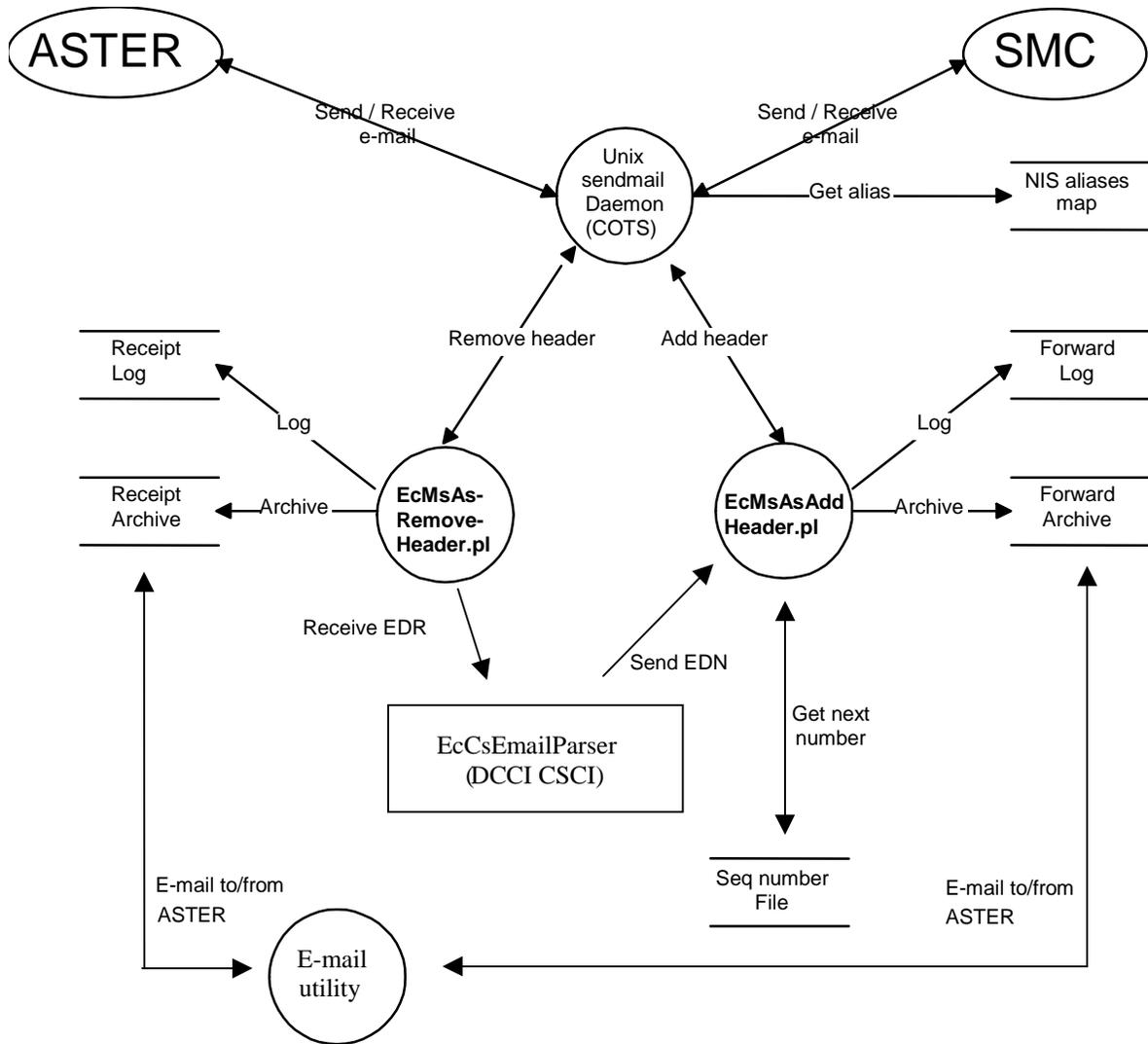


Figure 4.9.1.6.6.3-1. ASTER E-mail Header Handler Architecture Diagram

4.9.1.6.6.4 ASTER E-mail Header Handler Process Descriptions

Table 4.9.1.6.6.4-1 provides descriptions of the ASTER E-mail Header Handler processes shown in the ASTER E-mail Header Handler architecture diagram.

Table 4.9.1.6.6.4-1. ASTER E-mail Header Handler Processes

Process	Type	COTS / Developed	Functionality
Unix Sendmail daemon	Server	COTS	The sendmail daemon, which is provided with the UNIX operating system, handles delivery and receipt of e-mail messages.
EcMsAsAddHeader.pl	Other	Developed	This script is invoked by the sendmail daemon when a message is sent to an alias configured to process messages requiring ASTER e-mail headers before delivery. This perl script inserts the header into a message directed to the ASTER GDS.
EcMsAsRemoveHeader.pl	Other	Developed	This script is also invoked by the sendmail daemon when a message is sent to an alias configured to process e-mail containing ASTER e-mail headers. The perl script removes the header and forwards the message to the address defined in the alias.
E-mail utility	Other	COTS	The M&O Staff uses an e-mail utility for review of transmitted/received messages and the messages can be re-transmitted in the event of a problem.

4.9.1.6.6.5 ASTER E-mail Header Handler Process Interface Descriptions

Table 4.9.1.6.6.5-1 provides descriptions of the interface events shown in the ASTER E-mail Header Handler architecture diagram.

**Table 4.9.1.6.6.5-1. ASTER E-mail Header Handler Process Interface Events
(1 of 2)**

Interface	Event Frequency	Interface	Initiated By	Event Description
Send e-mail	One per e-mail send	SMC, ASTER GDS	Unix Sendmail daemon (API, system call or command line) – (COTS)	The UNIX sendmail daemon attempts to forward e-mail messages to the specified recipient at the SMC or the ASTER GDS.
Receive e-mail	One per e-mail receive	SMC, ASTER GDS	Unix Sendmail daemon (SMTP protocols) – (COTS)	The UNIX sendmail daemon receives and processes e-mail messages it has been configured to receive from the SMC and the ASTER GDS.
Get alias	One per get alias	<i>Data Store:</i> NIS aliases map	Unix Sendmail daemon (NIS system call) - [COTS]	While processing e-mail, the sendmail daemon checks to see if the specified recipient is a local user, an alias for another user, or an executable to stream the message into.
Add header	One per message	Unix Sendmail daemon [COTS]	<i>Script:</i> EcMsAsAddHeader.pl	The ASTER e-mail header is inserted in the body of an e-mail message by the EcMsAsAddHeader.pl script and sent to the Unix sendmail daemon to be sent to its destination.
Log	One per log	<i>Data Store:</i> Forward Log <i>Data Store:</i> Receipt Log	<i>Script:</i> EcMsAsAddHeader.pl <i>Script:</i> EcMsAsRemoveHeader.pl	An entry is added to note the e-mail message date, time, and recipient are being forwarded.

**Table 4.9.1.6.6.5-1. ASTER E-mail Header Handler Process Interface Events
(2 of 2)**

Interface	Event Frequency	Interface	Initiated By	Event Description
Archive	One per archive	<i>Data Store:</i> Forward Archive <i>Data Store:</i> Receipt Archive	<i>Script:</i> EcMsAsAddHeader.pl <i>Script:</i> EcMsAsRemoveHeader.pl	A copy of the e-mail message is stored in a format that can be read by Netscape mail (Setenv MAIL to the file location of the archive)
Send EDN	One per E-mail send	<i>Script:</i> EcMsAsAddHeader	<i>Process:</i> EcCsEmailParser <i>Class:</i> EcCsEmailParser	The EcCsEmailParser sends the EDN to the EcMsAsAddHeader to have a header added.
Get next number	One per get of next number	<i>Data Store:</i> Seq number File	<i>Script:</i> EcMsAsAddHeader.pl	The EcMsAcAddHeader.pl script obtains the next sequence number from a text file.
E-mail to/from ASTER	One per e-mail to/from ASTER	<i>Data Stores:</i> Receipt Archive, Forward Archive	<i>Process:</i> E-mail utility (e.g., Netscape)	E-mail messages with standard headers are sent to/from ECS users or M&O staff personnel from/to ASTER GDS users or operations personnel using SMTP protocols and copies are kept in data files within the ECS.
Receive EDR	One per EDR sent	<i>Process:</i> EcCsEmailParser <i>Class:</i> EcCsEmailParser	<i>Script:</i> EcMsAsRemoveHeader	After selecting the EDN, the ASTER GDS personnel send an EDR to the EcMsAsRemoveHeader, via the Unix sendmail daemon, to have the header removed.
Remove header	One per message	<i>Process:</i> EcCsEmailParser <i>Class:</i> EcCsEmailParser	<i>Script:</i> EcMsAsRemoveHeader.pl	The ASTER header is removed from messages sent for local delivery by the EcMsAsRemoveHeader.pl script and sent to the EcCsEmailParser.

4.9.1.6.6.6 ASTER E-mail Header Handler Data Stores

Table 4.9.1.6.6.6-1 provides descriptions of the data stores shown in the ASTER E-mail Header Handler architecture diagram.

Table 4.9.1.6.6.6-1. ASTER E-mail Header Handler Data Stores

Data Store	Type	Functionality
NIS aliases map	Other	This database provides the aliases used by sendmail to determine where to redirect the e-mail messages.
Seq number file	text file	This file contains the next available sequence number.
Receipt Log	text file	The date/time stamp and recipient are maintained in this log.
Forward Log	text file	The date/time stamp and recipient are maintained in this log.
Receipt Archive	text file	This file maintains copies of e-mail messages sent from the ASTER GDS.
Forward Archive	text file	This file contains copies of e-mail messages sent to the ASTER GDS.

4.9.2 Management Agent Computer Software Configuration Item Description

The Management Agent CSCI (MACI) is ECS developed and COTS software and consists of the SubAgent, Deputy Agent, Proxy Agent, and Master Agent.

The MACI is the interface between each DAAC's management platform and the ECS developed applications and COTS products distributed on hosts throughout the local area network. The MACI assures secure communications with ECS developed and COTS applications by using Remote Procedure Calls (RPCs) instead of the SNMP. The MACI implements the ECS application MIB that is a look-up table allowing communications between the management platform and subagents managing applications. The Operations Staff, to obtain management information about ECS applications, uses the ECS application MIB browser. The MACI accepts lifecycle commands from the management platform to start-up/shutdown applications and forwards application status to the management platform.

The ECS subagent communicates management requests and responses from the master agent and the Deputy agent to either an ECS developed application or to a COTS application via the Proxy Agent. It supports MIB extensions and performs local polling on host resources. The SubAgent, which is custom developed software built upon COTS libraries, makes the custom developed software remotely manageable.

4.9.2.1 Management Agent Functional Overview

The MACI manages and monitors ECS applications (via the CSS process framework managed servers and COTS applications). The Deputy agent handles secure delivery of requests for setting management information by using DCE remote procedure calls. The Proxy agent manages non-SNMP manageable COTS products. Its front-end has the MSS instrumentation software to communicate with the subagent. Its back-end communicates with the COTS. Each COTS process

provides a definition to the Proxy agents, which includes its start up and shutdown procedure description. The Proxy agent instantiates a manager object on behalf of each COTS process started. This manager object binds with and is monitored by the subagent. The Master agent is an SNMP agent that manages resource distribution to one or more subagents using a client/server communications paradigm. The communication between the client and the server is encapsulated in the SNMP Multiplexing (SMUX) protocol. When a sub-agent is started, it connects with the master agent running on the host. If the connection is successful, the subagent registers the branch of the MIB it is managing with the master agent.

The Encapsulator is a specialized sub-agent that enables incompatible and non-extensible SNMP agents to be present on the same processor as the Master Agent. These SNMP agents are vendor supplied agents installed on the workstation as part of the Operating System.

It is assumed that the master agent and the subagent, responsible for the management of applications, are always running. They are started when the host is booted. In addition, Tivoli monitors the status of the subagent and attempts to restart it if it is not running. The request to start or shutdown an application can be issued on the management application. This request is passed securely (using DCE remote procedure calls instead of SNMP messages) to the subagent on remote hosts and the startup or shutdown actions are performed. The life cycle services (i.e., startup, shutdown, and other requests) trigger event notifications to the MSS enterprise and management framework. These requests are bundled within RPCs by the HP OpenView custom software and sent to the remote subagent's Deputy Gate.

The MSS requires each managed host, standard MIB, Host Resource MIB, and the network device MIBs to be supported by vendor agents. In addition, a managed object model is defined by the MSS for ECS applications in the SNMP MIB format as the ECS application MIB. The management agent service implements the ECS application MIB. The MIB information is composed of different types of attributes: configuration, performance, fault, dynamic, static, and traps.

Management applications can make SNMP requests to retrieve management information as MIB values. They can also set certain management information via secure DCE RPCs.

4.9.2.2 Management Agent Context

Figure 4.9.2.2-1 is the Management Agent CSCI (MACI) context diagram. The diagram shows the events sent to the MACI and the events the MACI sends to the HP OpenView COTS product. Table 4.9.2.2-1 provides descriptions of the interface events shown in the MACI context diagram.

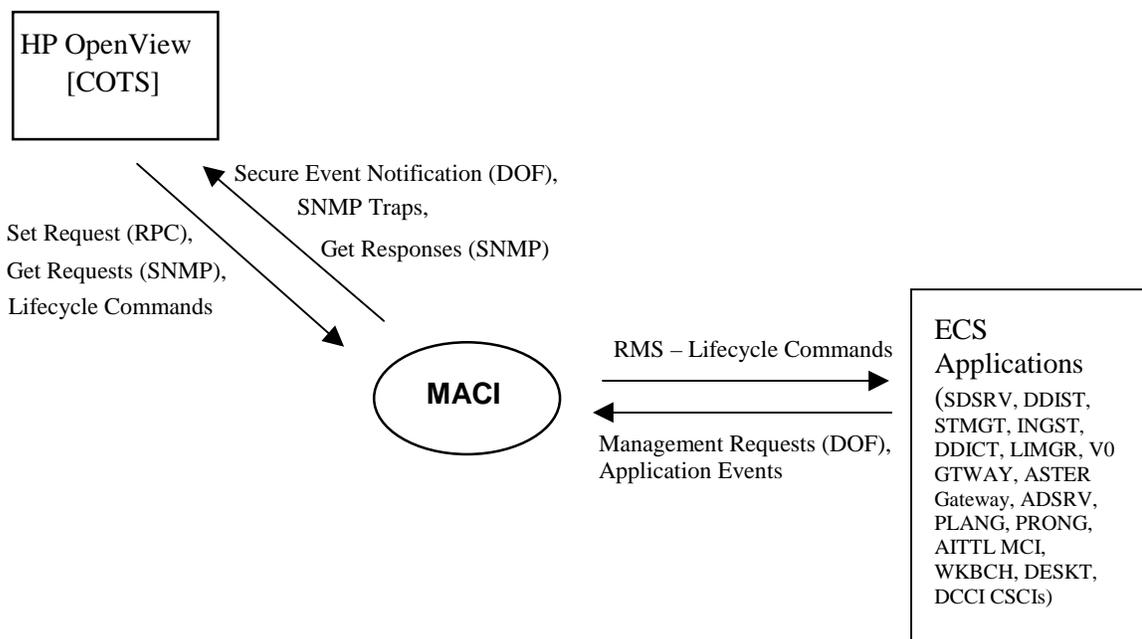


Figure 4.9.2.2-1. Management Agent CSCI (MACI) Context Diagram

Table 4.9.2.2-1. Management Agent CSCI (MACI) Interface Events (1 of 2)

Event	Interface Event Description
Request Management Services	The MSS provides a basic management library of services to the subsystems, implemented as client or server applications, using the CSS Process Framework. The basic management library of services include: <ul style="list-style-type: none"> Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training).
Management Requests (DOF)	When an ECS application is started, it sends a request to the Registry service in the SubAgent to start monitoring it.
Application Events	Errors logged by ECS applications are sent to the ALOG file. Application events can also be generated by the subagent in response to topology changes such as a process startup/shutdown. Finally, application events include the obtaining and registering (sending the subagent) to the performance, configuration, and fault metrics of the particular application.
Set Request (RPC)	The MIB on the MSS Server sends Set Requests to the sub Agent (via remote procedure calls) running on the remote host.
Get Requests (SNMP)	The HP OpenView management platform sends the Get requests to the Peer Master Agent running on the MSS server or ECS host. The SNMP protocol is used to send the requests.

Table 4.9.2.2-1. Management Agent CSCI (MACI) Interface Events (2 of 2)

Event	Interface Event Description
Lifecycle Commands	HP OpenView issues startup/shutdown lifecycle commands via the MSS management subagent to applications in the managed hosts.
Secure Event Notification (DOF)	The M&O staff send all application events sent from ECS applications to the SubAgent to the HP OpenView COTS for display and action, if necessary.
SNMP Traps	The Deputy Agent converts the application events received from the subagent and converts the events into traps. These traps are forwarded to the Trapd daemon. The Trapd logs the traps into the Trapd log where HP OpenView (HPOV) reads them. For example, if an application dies, the subagent sends a topology change event to the Deputy Agent. The Deputy Agent converts the change event into a trap and forwards the trap to HPOV, and the corresponding icon read the trap on the GUI turns red to indicate the application died.
Get Responses (SNMP)	The Peer Master Agent returns the responses for Get requests from the Subagent to the HP OpenView management platform.

4.9.2.3 Management Agent Architecture

Figure 4.9.2.3-1 is the Management Agent CSCI (MACI) architecture diagram. The diagram shows the events sent to the MACI management agent processes and the events the MACI management agent processes send to each other, other CSCIs and COTS products.

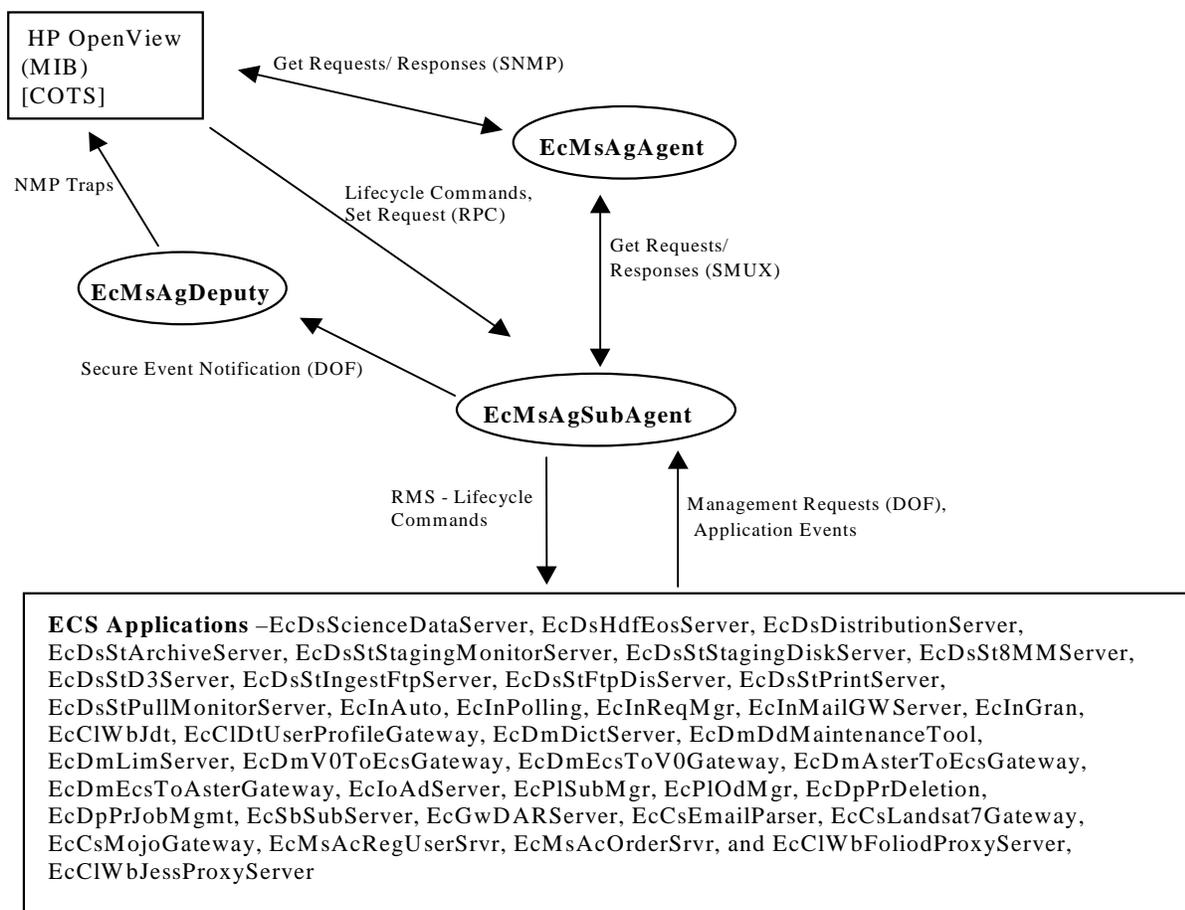


Figure 4.9.2.3-1. Management Agent CSCI (MACI) Architecture Diagram

4.9.2.4 Management Agent Process Descriptions

Table 4.9.2.4-1 provides descriptions of the processes shown in the MACI architecture diagram.

Table 4.9.2.4-1. Management Agent CSCI (MACI) Processes (1 of 2)

Process	Type	COTS / Developed	Functionality
EcMsAgSubAgent	Other	COTS/Developed	The EcMsAgSubAgent manages startup, shutdown, and monitoring of ECS developed and COTS applications.
EcMsAgDeputy	Other	COTS/Developed	The Deputy Agent receives application events from the EcMsAgSubAgent in the CDS cell, converts the application events into traps and forwards the traps to the HP Open View COTS package.

Table 4.9.2.4-1. Management Agent CSCI (MACI) Processes (2 of 2)

Process	Type	COTS / Developed	Functionality
EcMsAgAgent	Other	COTS	The Master Agent is an SNMP agent that manages the distribution of application management requests to the EcMsAgSubAgent on each host.

4.9.2.5 Management Agent Process Interface Descriptions

Table 4.9.2.5-1 provides descriptions of the interface events shown in the MACI architecture diagram.

Table 4.9.2.5-1. Management Agent CSCI (MACI) Process Interface Events (1 of 5)

Event	Event Frequency	Interface	Initiated By	Event Description
Get Requests/ Responses (SMUX)	One per get of requests/responses	<i>Process:</i> EcMsAgSubAgent (responses) <i>Class:</i> MsAgPortMonitor	<i>Process:</i> EcMsAgAgent (request) <i>Class:</i> MsAgPortMonitor	The EcMsAgSubAgent connects to the EcMsAgAgent when it starts. If successful, the EcMsAgSubAgent registers the root of the ECS MIB with the EcMsAgAgent, and begins to monitor for ECS MIB requests. If not successful, the EcMsAgSubAgent periodically tries to connect to the EcMsAgAgent and logs error messages indicating the connection failure until a successful connection takes place. All SNMP Get Requests intended for receipt by the ECS MIB are sent to the EcMsAgSubAgent by the EcMsAgAgent, and responses are returned from the EcMsAgSubAgent to the EcMsAgAgent which formulates an SNMP response to the management platform.

Table 4.9.2.5-1. Management Agent CSCI (MACI) Process Interface Events (2 of 5)

Event	Event Frequency	Interface	Initiated By	Event Description
Management Requests (DOF)	One per management request	<i>Process:</i> EcMsAgSubAgent <i>Classes:</i> MsAgRegistry, MsAgEventHandler	<i>Processes:</i> ECS Applications (as identified in the architecture diagram)	When an ECS application is started, the application sends a request to the Registry service in the EcMsAgSubAgent to start monitoring it.
Application Events	One per application event	<i>Process:</i> EcMsAgSubAgent <i>Classes:</i> EcAgEvent, MsAgPerfEvent, MsAgEventMgr	<i>Processes:</i> ECS Applications (as identified in the architecture diagram) Library: EcPf <i>Classes:</i> EcPfManagedServer, EcPfclient	ECS Applications use the Process Framework Library to send application events to the EcMsAgSubAgent and includes events contained in the system log (such as too many logins).

Table 4.9.2.5-1. Management Agent CSCI (MACI) Process Interface Events (3 of 5)

Event	Event Frequency	Interface	Initiated By	Event Description
Request Management Services	One command per request	<p><i>Processes:</i> EcDsScienceDataServer, EcDsHdfEosServer, EcDsDistributionServer, EcDsStArchiveServer, EcDsStStagingMonitorServer, EcDsStStagingDiskServer, EcDsSt8MMServer, EcDsStD3Server, EcDsStIngestFtpServer, EcDsStFtpDisServer, EcDsStPrintServer, EcDsStPullMonitorServer, EcInAuto, EcInPolling, EcInReqMgr, EcInEmailGWServer, EcInGran, EcCIWbJdt, EcCIDtUserProfileGateway, EcDmDictServer, EcDmMaintenanceTool, EcDmLimServer, EcDmV0ToEcsGateway, EcDmEcsToV0Gateway, EcDmAsterToEcsGateway, EcDmEcsToAsterGateway, EcIoAdServer, EcPISubMgr, EcPIOdMgr, EcDpPrDeletion, EcDpPrJobMgmt, EcSbSubServer, EcGwDARServer, EcCsEmailParser, EcCsLandsat7Gateway, EcCsMojoGateway, EcMsAcRegUserSrvr, EcMsAcOrderSrvr, EcCIWbFoliodProxyServer, EcCIWbJessProxyServer</p>	<p><i>Process:</i> EcMsAgSubAgent</p> <p><i>Library:</i> EcAgInstrm</p> <p><i>Class:</i> EcAgManager</p>	<p>The EcMsAgSubAgent provides a basic management library of services to the processes, implemented as client or server applications, using the DCCI CSCI Process Framework. The basic management library of services include:</p> <ul style="list-style-type: none"> <p>Lifecycle Commands - The HPOV Network Node Manager forwards (via DCE RPCs), to the MSS Sub Agent running on each managed host, requests to start and stop ECS applications. A start request has mode and temperature parameters that the MSS Sub Agent uses in constructing its command line startup request. Stop requests precipitate a PF shutdown RPC call to the target ECS application from the MSS Sub Agent. Managed applications use the application interface PFGETMODE to obtain their operational mode (e.g., Ops, test, or training).</p>

Table 4.9.2.5-1. Management Agent CSCI (MACI) Process Interface Events (4 of 5)

Event	Event Frequency	Interface	Initiated By	Event Description
Secure Event Notification (DOF)	One per secure event notification	<i>Process:</i> EcMsAgDeputy <i>Classes:</i> MsAgEventManager, EcAgEvent	<i>Process:</i> EcMsAgSubAgent <i>Classes:</i> MsAgEventManager, EcAgEvent	All the events from the EcMsAgSubAgent are sent to a Deputy Agent via a RPC.
SNMP Traps	One per SNMP trap	<i>Process:</i> HP OpenView (COTS)	<i>Process:</i> EcMsAgDeputy <i>Classes:</i> MsAgDeputyGate, MsAgSnmpPdu	The EcMsAgDeputy converts the application events received from the EcMsAgSubAgent into traps. The traps are forwarded to the Trapd daemon. Trapd logs the traps into the Trapd log where they are read by HPOV. For example, if an application dies, the EcMsAgSubAgent sends a topology change event to the Deputy Agent. The EcMsAgDeputy converts the change event into a trap and forwards the trap, which is read by HPOV. HPOV causes the corresponding icon on the GUI to turn red to indicate the application has died.
Get Requests/ Responses (SNMP)	One per get request and response	<i>Process:</i> EcMsAgAgent (response) <i>Class:</i> MsAgAgent (COTS)	<i>Process:</i> HP OpenView (COTS) (request)	The HP OpenView management platform sends the Get requests to the EcMsAgAgent running on the remote host. The SNMP protocol is used to send the get requests. Responses?

Table 4.9.2.5-1. Management Agent CSCI (MACI) Process Interface Events (5 of 5)

Event	Event Frequency	Interface	Initiated By	Event Description
Lifecycle Commands	One per lifecycle command	<i>Process:</i> EcMsAgSubAgent <i>Class:</i> MsAgDeputyGate	<i>Process:</i> HP OpenView (COTS)	HP OpenView sends startup/shutdown lifecycle commands to the EcMsAgSubAgent. The EcMsAgSubAgent starts servers via a system call as specified in the server's *.ACFG or *.PCFG files. The EcMsAgSubAgent uses the CSS Process Framework Library calls to shutdown running servers.
Set Request (RPC)	One per request	<i>Process:</i> EcMsAgSubAgent <i>Class:</i> MsAgDeputyGate	<i>Process:</i> EcMsCmMibBwsr <i>Library:</i> DCE	The EcMsCmMibBwsr on the HPOV Network Node Manager sends Set Requests to the EcMsAgSubAgent (via remote procedure calls) running on the remote host.

4.9.2.6 Management Agent Data Stores

Data Stores are not applicable for the Management Agent CSCI.

4.9.3 Management Logistics Computer Software Configuration Item Description

4.9.3.1 Management Logistics Functional Overview

The Management Logistics CSCI (MLCI) supports the configuration management of the ECS. The MLCI is the following six CSCs:

- **Baseline Manager (BLM):** The BLM CSC maintains records of baselined, operational configurations. The BLM CSC identifies hardware/software items, sites, versions, interdependencies, and tracks change history.
- **Inventory, Logistics, Maintenance (ILM) Manager:** The ILM CSC maintains records on contract purchased items containing information such as vendor, date of receipt, installation, and warranty expiration. The ILM CSC also maintains maintenance records about contract purchased items.
- **Software Change Manager:** The Software Change Manager CSC supports maintenance and change control of the science software configuration at each DAAC.

- Change Request Manager: The Change Request Manager CSC tracks and maintains Configuration Change Requests (CCRs) at each DAAC.
- Software Distribution Manager: The Software Distribution Manager CSC supports the distribution of ECS software, database information, software documentation, and COTS files to other various ECS destinations.
- Software License Manager: The Software License Manager CSC monitors and controls licensing of COTS products installed in the ECS.

4.9.3.2 Management Logistics Context

Figure 4.9.3.2-1 is the Management Logistics CSCI (MLCI) context diagram. The diagram shows the events sent to the MLCI and the events the MLCI sends to other CSCIs or CSCs. Table 4.9.3.2-1 provides descriptions of the interface events shown in the MLCI context diagram.

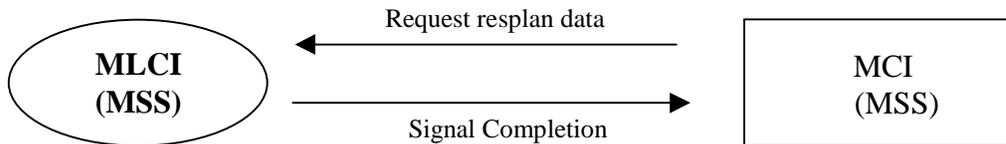


Figure 4.9.3.2-1. Management Logistics CSCI (MLCI) Context Diagram

Table 4.9.3.2-1. Management Logistics CSCI (MLCI) Interface Events

Event	Interface Event Description
Request resplan data	<p>The MLCI receives a request from the MCI for data about hosts, hardware, software, disk partitions, and strings of hosts. This request constitutes the site's production baseline as of a specified date and is flagged in XRP-II as "planning resources." Arguments associated with the request are the baseline date and a code used by MCI to notify Resource Planners of the outcome of the request. The request format is:</p> <p style="padding-left: 40px;">resplan <mmddy> <code></p> <p>In response, a set of ASCII records containing a header record followed by one or more detail records for each qualifying baseline. The header record contains a text message identifying the production baseline specified and the number of data records returned for it. Detail records describe the items marked as "planning resource" in the Baseline Manager database that constitute the site's production baseline as of an operator-specified date. Data in a detail record is separated by a pipe symbol " " and varies by type of item as follows:</p> <ul style="list-style-type: none"> • host items - "host", name, description, control item id, status, install date, # CPUs, total RAM, processing string name, string's control item id • hardware items - "hardware", name, description, control item id, status, install date • software items - "software", description, version, control item id, status, install date, associated host name, host control item id • disk partition items - "partition", device name, directory name, control item id, status, install date, partition size, block size, logical allocation, associated host's name, host's control item id • processing string items - "string", control item id, name, description, status, install date
Signal Completion	<p>A notification from the MLCI to inform the MCI that the resplan data request has been processed. The notice is made via Tivoli's "wasync" utility. It contains:</p> <ul style="list-style-type: none"> • a code indicating the purpose of the notice • an associated informational message

4.9.3.3 Management Logistics Architecture

An architecture diagram does not apply to the MLCI because it consists of standalone tools.

4.9.3.4 Management Logistics Process Description

Process descriptions are not applicable for the MLCI.

4.9.3.5 Management Logistics Process Interface Descriptions

Process Interface Descriptions are not applicable for the MLCI.

4.9.3.6 Management Logistics Data Stores

Data Stores are not applicable for the MLCI.

4.9.3.6.1 MLCI - Baseline Manager Computer Software Component Description

4.9.3.6.1.1 Baseline Manager Functional Overview

Baseline Manager aids the DAACs, EOC, and SMC staffs in maintaining records that describe what comprises baselined, operational system configurations. These records primarily identify the makes/models and versions of hardware and software items the baselines contain, and they can identify item interdependencies and the sites where baseline items are deployed. Additionally, the records track subsystems and networks, while maintaining a history of changes and traceability of version-controlled items to their predecessors and associated system releases. Baseline Manager at the DAACs and EOC maintains records about baselines deployed at the site. At the SMC, Baseline Manager maintains records about baselines system-wide. A COTS application called XRP II is used to accomplish baseline management tasks.

- The functionality for the Baseline Manager is implemented via the COTS products XRP-II and ACCELL, plus the following to supplement what is provided by the vendor: Instructions for installing and configuring XRP-II and ACCELL in an ECS environment
- XRP specifications and definitions for generating the following ECS reports:
 - Hardware-Software Map – A list of software control items in a specified baseline as of a specified date
 - Hardware-Patch Map – A list of the patches and patch bundles in a specified baseline as of a specified date
 - Hardware Map – A list of hardware control items in a specified baseline as of a specified date
 - COTS Software Version Baseline – Descriptions of the software control items in a specified baseline as of a specified date
 - Patch Baseline Report – Descriptions of the patches in a specified baseline as of a specified date
 - Site-Host Map – A matrix presenting the names of the ECS hosts performing corresponding functions across the various sites for a specified baseline as of a specified date
 - Baselined Documents (Title Order) Report – A list of documents sorted by title for a specified baseline as of a specified date
 - Baselined Documents (Number Order) Report – A list of documents sorted by number for a specified baseline as of a specified date

4.9.3.6.1.2 Baseline Manager Context Diagram

Baseline Manager runs at the SMC, EOC, and each DAAC. Baseline records can be exchanged among sites via formatted data files. These files are created locally on demand and transferred to

other sites as appropriate via the ftp service where Operations Staff uses them to update their site's database.

Baseline Manager has one interaction with another MSS CSCI, namely MCI. As shown in Figure 4.9.3.6.1.2-1, the Baseline Manager provides select baseline data records and an associated end-of-task signal to MCI (Tivoli) in response to Resource Planner (resplan) data requests. Table 4.9.3.6.1.2-1 provides descriptions of the interface events shown in the Baseline Manager context diagram.

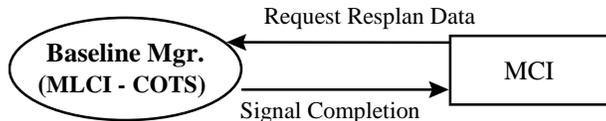


Figure 4.9.3.6.1.2-1. Baseline Manager Context Diagram

Table 4.9.3.6.1.2-1. Baseline Manager Interface Events (1 of 2)

Event	Interface Event Description
Request Resplan Data	<p>Request to XRP-II for data about hosts, hardware, software, disk partitions and strings of hosts. The request constitutes the site's production baseline as of a specified date and is flagged in XRP-II as "planning resources." Arguments associated with the request are the baseline date and a code used by MCI to notify Resource Planners of the outcome of the request. The request format is:</p> <p style="padding-left: 40px;">resplan <mmddyy> <code></p> <p>In response, a set of ASCII records containing a header record followed by one or more detail records is returned for each qualifying baseline. The header record contains a text message identifying the production baseline specified and the number of data records returned for it. Detail records describe the items marked as "planning resource" in the Baseline Manager database that constitute the site's production baseline as of an operator-specified date. Data in a detailed record is separated by a pipe symbol " " and varies by type of item as follows:</p> <ul style="list-style-type: none"> • host items - "host", name, description, control item id, status, install date, # CPUs, total RAM, processing string name, string's control item id • hardware items - "hardware", name, description, control item id, status, install date • software items - "software", description, version, control item id, status, install date, associated host name, host control item id • disk partition items - "partition", device name, directory name, control item id, status, install date, partition size, block size, logical allocation, associated host's name, host's control item id • processing string items - "string", control item id, name, description, status, install date

Table 4.9.3.6.1.2-1. Baseline Manager Interface Events (2 of 2)

Event	Interface Event Description
Signal Completion	Notification for MCI that the resplan data request has been processed. The notice is made via Tivoli's "wasync" utility. It contains: <ul style="list-style-type: none"> • a code indicating the purpose of the notice • an associated informational message.

4.9.3.6.1.3 Baseline Manager Architecture

Baseline Manager is implemented as a specially configured version of the commercially available manufacturing management system "XRP-II". It is a single, standalone, non-client/server application with an internal relational database. XRP-II uses the UNIFY relational database management system marketed as part of the ACCELL Integrated Development System product. The database (refer to Figure 4.9.3.6.1.3-1) is shared with the Inventory/Logistics/Maintenance (ILM) Manager CSC, which is also implemented using XRP-II. Data records can be exported via formatted data files for use by Baseline Manager applications at other sites.

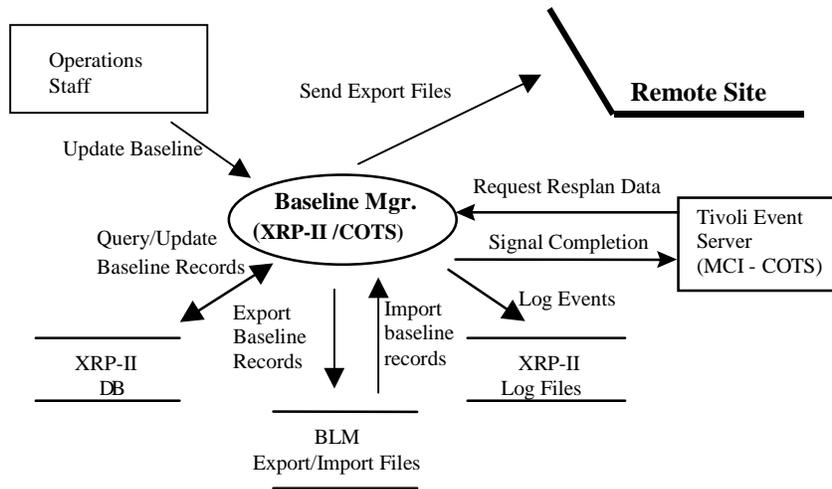


Figure 4.9.3.6.1.3-1. Baseline Manager Architecture Diagram

Baseline Manager's controlling program is the XRP-II menu handler. Invoked via the startup script "pcs", the menu handler uses the Operations staff member's userid to present a character-based user interface (CHUI) with menus, data entry screens, and permissions for functions and menus the Operations staff are authorized to use.

4.9.3.6.1.4 Baseline Manager Process Descriptions

The XRP-II menus and screens, which invoke sub-processes that perform functions, are summarized in Table 4.9.3.6.1.4-1.

Table 4.9.3.6.1.4-1. Baseline Manager Processes

Process	Type	COTS / Developed	Functionality
Baseline Mgr (XRP-II)	Other	COTS	<ul style="list-style-type: none"> • Manages records identifying deployed ECS baselines and the versions of hardware and software items the baselines contain. • Maintains chronological histories of baseline changes. • Maintains item traceability to predecessors and associated system releases. • Exports/imports records using formatted data files to support data exchange between the DAACs and the SMC. • Generates both pre-defined and ad hoc reports about baseline-related data

4.9.3.6.1.5 Baseline Manager Process Interface Descriptions

Baseline Manager’s sole ECS processing interface is with the MCI’s Tivoli application. When commanded by resource planners, Tivoli issues Baseline Manager a request for resource planner data. Tivoli makes the request by running the Baseline Manager’s “resplan” script, passing a date and code as arguments. Baseline Manager responds by producing a set of formatted data records and a signal containing a status message. The data records are written to stdout, which Tivoli reads and routes to a predetermined path name and host as part of Tivoli’s configuration. The signal, sent to Tivoli via its “wasync” command, passes an action code and a message which Tivoli can use for notifying Resource Planners (e.g., in a pop-up window) that the data records have been delivered. Table 4.9.3.6.1.5-1 provides descriptions of the interface events shown in the Baseline Manager architecture diagram.

Table 4.9.3.6.1.5-1. Baseline Manager Process Interface Events (1 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
Request Resplan Data	One per resplan data request	<i>Program:</i> Baseline Mgr. (XRP-II /COTS) <i>Script:</i> resplan (XRP-II)	Process: Tivoli Event Server (COTS)	Request to XRP-II for data about hosts, hardware, software, disk partitions and strings of hosts. The request constitutes the site's production baseline as of a specified date and is flagged in XRP-II as "planning resources." Arguments associated with the request are the baseline date and a code used by MCI to notify Resource Planners of the outcome of the request. The request format is: <pre>r esplan <mmddy> <code></pre> In response, a set of ASCII records containing a header record followed by one or more detail records is returned for each qualifying baseline. The header record contains a text message identifying the production baseline specified and the number of data records returned for it. Detail records describe the items marked as "planning resource" in the Baseline Manager database that constitute the site's production baseline as of an operator-specified date. Data in a detailed record is separated by a pipe symbol " " and varies by type of item as follows: <ul style="list-style-type: none"> • host items - "host", name, description, control item id, status, install date, # CPUs, total RAM, processing string name, string's control item id • hardware items - "hardware", name, description, control item id, status, install date • software items - "software", description, version, control item id, status, install date, associated host name, host control item id • disk partition items - "partition", device name, directory name, control item id, status, install date, partition size, block size, logical allocation, associated host's name, host's control item id • processing string items - "string", control item id, name, description, status, install date

Table 4.9.3.6.1.5-1. Baseline Manager Process Interface Events (2 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
Signal Completion	One per resplan data request	Process: Tivoli Event Server (COTS) wasync utility program	<i>Program:</i> Baseline Manager (XRP-II /COTS) <i>Script:</i> resplan	Notification for MCI that the resplan data request has been processed. The notice is made via Tivoli's "wasync" utility. It contains: <ul style="list-style-type: none"> • a code indicating the purpose of the notice • an associated informational message
Log Events	One per event recorded	<i>Data Store:</i> XRP-II Log Files	<i>Program:</i> Baseline Manager (XRP-II /COTS)	Stores records describing notable events as described in Table 4.9.3.6.1.6-1 under data store named "XRP-II log files".
Import Baseline Records	One per data import request	<i>Data Store:</i> BLM Import/Export Files	<i>Program:</i> Baseline Manager CHUI (XRP-II menu and data entry screens)	Loads exported data records received from other sites.
Export Baseline Records	One per data export request	<i>Data Store:</i> BLM Import/Export Files	<i>Program:</i> Baseline Manager CHUI (XRP-II menu and data entry screens)	Extracts and tars data records for use at other sites.
Query/Update Baseline Records	One per item created, deleted, modified, or retrieved	<i>Data Store:</i> XRP-II DB	<i>Program:</i> Baseline Manager CHUI (XRP-II menu and data entry screens)	Maintains records of baselined configuration items for ECS. Tracks the identity of all devices and software items. Maintains a change history for all baselined items.
Update Baseline	One per request to update baseline	<i>Program:</i> Baseline Manager CHUI (XRP-II menu and data entry screens) <i>Script:</i> Pcs	Operations Staff Command Line Interface	Enables operations staff to: <ul style="list-style-type: none"> • identify/catalog items, which are in the baseline • identify assemblies of hardware and software items that form operational system configurations • identify hardware and software interdependencies • keep a chronological history of baseline changes • trace version-controlled items to previous versions • exchange baseline data records across sites

Table 4.9.3.6.1.5-1. Baseline Manager Process Interface Events (3 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
Send Export Files	One per data export request	<i>Data Store:</i> BLM Import/Export Files	Operations Staff <i>Program:</i> Baseline Manager (XRP-II /COTS)	If requested by the Operations Staff when exporting baseline records, transfers a tar file containing the exported records to one or more sites via the FTP service.

4.9.3.6.1.6 Baseline Manager Data Stores

Baseline Manager’s principal data stores are the XRP-II database, log files, and formatted data files used for exporting and importing Baseline Manager records. The data store descriptions are provided in Table 4.9.3.6.1.6-1.

Table 4.9.3.6.1.6-1. Baseline Manager Data Stores (1 of 2)

Data Store	Type	Description
XRP-II DB	database	A non-replicated collection of baseline, inventory, and maintenance-related data that exists at each site. For Baseline Manager, it principally contains records identifying and describing: <ul style="list-style-type: none"> • control items: version-controlled entities such as baselines, software products, hardware devices, and documents • product structures: parent/product component pairings defining the ingredients – or bill of material – for control item assemblies • engineering change notices: mechanisms by which configuration changes with their effectivity dates are defined for an assembly • control item implementation status – mappings of control items against deployment sites, together with status and date for each • control item interdependencies – specified dependencies that any control item has on another
BLM export/import files	tar file	Formatted data files created as necessary to exchange Baseline Manager records among sites. Each contains either: <ul style="list-style-type: none"> • all site-unique Baseline Manager records new or changed at a site since the previous export of changed site-unique records • all “core” Baseline Manager records changed since the previous export of changed “core” records • all the Baseline Manager records for a specified system release, baseline, or other configuration-controlled item or assembly • all records dumped from one or more XRP-II database tables

Table 4.9.3.6.1.6-1. Baseline Manager Data Stores (2 of 2)

Data Store	Type	Description
XRP-II log files	text files	A collection of files containing information about XRP-II events and errors encountered during processing including: <ul style="list-style-type: none"> • xrp.log - userid, date/time, and result of operator attempts to log into XRP-II • datadump.log - userid, date/time, and result of operator attempts to dump XRP-II data in bulk into ASCII files • dataread.log - used to load XRP-II data in bulk from ASCII files • errlog and *.err files - details about fatal errors; useful mainly to XRP-II programmers • import.log - events associated with importing data from other sites
	binary file	<ul style="list-style-type: none"> • unify transaction log – A collection of records for journaling and rolling forward database transactions. Used in conjunction with database backups and restores

4.9.3.6.2 MLCI - Inventory/Logistics/Maintenance Manager Computer Software Component Description

4.9.3.6.2.1 Inventory/Logistics/Maintenance Manager Functional Overview

The Inventory/Logistics/Maintenance (ILM) Manager is the principal tool used for ECS integrated logistics support (ILS). It tracks and maintains the key data pertaining to ECS contract purchased equipment including hardware, COTS software, COTS documentation (hardware and software), spares and consumable items, and Government Furnished Equipment (GFE). Information tracked for each item includes dates (e.g., receipt, installation, and warranty expiration), user, location, manufacturer, vendor, purchase order, Original Equipment Manufacturer (OEM) part number, model/version, and description. The ILM Manager also stores and maintains detailed maintenance data on hardware, to the item level, including preventive and corrective maintenance.

The ILM Manager gives authorized users access to property data via a character-based user interface (CHUI). The user can select individual records, or sets of records, by entering one or more valid attributes in a data entry screen. Reporting capabilities are available to the user by accessing the report menu and entering valid values. The user can also obtain ad-hoc reports from any screen output, including table based, form based and delimited ASCII formats to screen, files, or printers.

For maintenance of contract hardware and COTS software, the ILM Manager tracks the OEM warranty expiration dates, maintenance contract, phone numbers, contact, maintenance password and license key data.

The functionality for the Inventory/Logistics/Maintenance Manager is implemented via the COTS products XRP-II and ACCELL, the same applications used for baseline management (see

section 4.9.3.6.1). No custom scripts are used in this CSC. However, the following supplements what is provided by the vendor:

- Instructions for installing and configuring XRP-II and ACCELL in an ECS environment

4.9.3.6.2.2 Inventory/Logistics/Maintenance Manager Context

The ILM Manager does not have an interface with any other subsystem, CSCIs or CSCs.

4.9.3.6.2.3 Inventory/Logistics/Maintenance Manager Architecture

The ILM Manager is implemented as a specially configured version of the commercially available manufacturing management system “XRP-II”. It is a single, standalone, non-client/server application with an internal relational database. XRP-II uses the UNIFY relational database management system marketed as part of the ACCELL Integrated Development System product. The database is shared with the Baseline Manager CSC. Data records can be exported via formatted data files for use by the ILM Manager applications at other sites. Figure 4.9.3.6.2.3-1 is the ILM Manager architecture diagram.

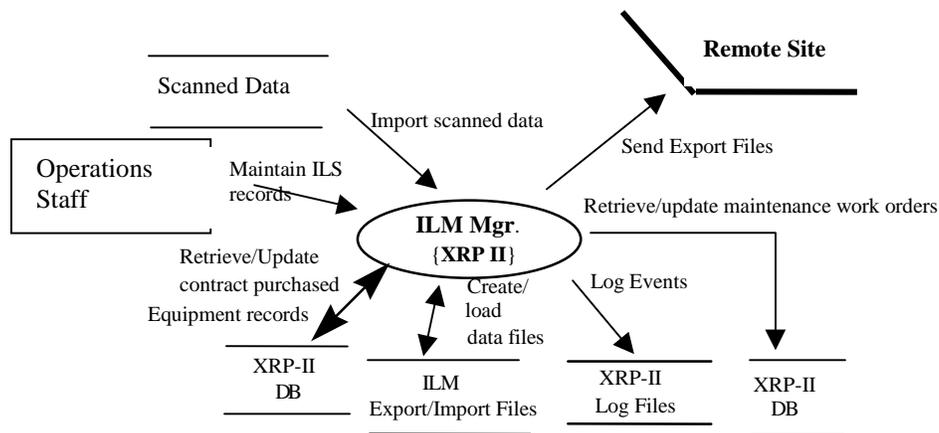


Figure 4.9.3.6.2.3-1. Inventory/Logistics/Maintenance (ILM) Manager Architecture Diagram

4.9.3.6.2.4 Inventory/Logistics/Maintenance Manager Process Descriptions

ILM’s controlling program is XRP’s menu handler. Invoked via the startup script “ilmusr”, the menu handler uses the operator’s userid to present a character-based interface with menus, data entry screens, and permissions for functions and menus the operators are authorized to use. The menus and screens in turn invoke sub-processes that perform functions summarized in Table 4.9.3.6.2.4-1.

Table 4.9.3.6.2.4-1. Inventory/Logistics/Maintenance (ILM) Manager Processes

Process	Type	COTS / Developed	Functionality
ILM Mgr. (XRP-II)	Other	COTS	<p>The ILM performs the following tasks:</p> <ul style="list-style-type: none"> • Captures and maintains all pertinent data for project hardware and COTS software • Manages, distributes, maintains reorder thresholds, and reports on consumables and spares • Manages, controls and reports on preventative maintenance actions • Manages, controls and reports on maintenance items other than preventative maintenance actions • Maintains historical log of maintenance actions against individual items within ILM • Tracks movement and archive actions for project property and reports on same • Manages and controls receipts against purchase orders • Maintains all other pertinent property management data required for the efficient use of the ILM tool such as vendor, buyer, user, manufacturer, and internal usage codes • Imports scanned inventory of items recorded by a bar code scanner. Reconciles scanned inventory with the XRP-II database

4.9.3.6.2.5 Inventory/Logistics/Maintenance Manager Process Interfaces

Table 4.9.3.6.2.5-1 provides descriptions of the interface events shown in the ILM Manager architecture diagram.

Table 4.9.3.6.2.5-1. ILM Manager Process Interface Events (1 of 2)

Event	Event Frequency	Interface	Initiated By	Event Description
Send Export Files	One per data export request	<i>Data Store:</i> ILM Import/Export Files	<i>Program:</i> ILM Mgr. (XRP-II)	If requested by the Operations Staff when exporting baseline records, transfers a tar file containing exported baseline records to one or more sites, via the FTP service.
Retrieve/update maintenance work orders	Once per retrieval or update	<i>Data Store:</i> XRP-II DB	<i>Program:</i> ILM Mgr. (XRP-II)	Maintains and retrieves information for maintenance work orders.

Table 4.9.3.6.2.5-1. ILM Manager Process Interface Events (2 of 2)

Event	Event Frequency	Interface	Initiated By	Event Description
Log events	Once per update	<i>Data Store:</i> XRP-II Log Files	<i>Program:</i> ILM Mgr. (XRP-II)	Logs information on activities performed by this COTS package. Retrieves and sends log files to the SMC for long term archival.
Create/Load data files	Once per creation or retrieval	<i>Data Store:</i> ILM Export/Import Files	<i>Program:</i> ILM Mgr. {XRP-II}	Maintains data files of contract purchased equipment from other sites (DAACs, EOC, and SMC) and sends information to other sites for spare and consumable parts information as well as for backup/retrieve purposes.
Retrieve/Update contract purchased equipment records	Once per retrieval or update	<i>Data Store:</i> XRP-II DB	<i>Program:</i> ILM Mgr. (XRP-II)	Maintains and retrieves contract purchased equipment information for ECS hardware, COTS software, COTS documentation, spare parts, consumable items such as printer ribbons and GFE. Tracks warranties, licenses, and maintenance for equipment.
Maintain ILS records	Once per request to maintain contract purchased equipment or maintenance work order records	<i>Program:</i> ILM Manager (XRP-II)	Operations Staff <i>Program:</i> ILM Manager (XRP-II) CHUI	Enables the Operations Staff to: <ul style="list-style-type: none"> • Identify inventory items, including spares and consumables • Track receipt, transfer, installation, condition, and disposition of inventory items • Track maintenance contracts and actions for inventory items • Record data needed for RMA analyses
Import scanned data	Once per import	<i>Data Store:</i> Scanned Data	<i>Program:</i> ILM Mgr. (XRP-II)	Imports scanned data obtained by the bar code scanner for processing against the XRP-II database.

4.9.3.6.3 MLCI - Software Change Manager Computer Software Component Description

4.9.3.6.3.1 Software Change Manager Functional Overview

The Software Change Manager aids the DAACs, EOC, and SMC staffs in organizing and partitioning software, controlling software changes and versions, and in assembling sets of software for release purposes. The Software Change Manager consists of a COTS application called ClearCase.

4.9.3.6.3.2 Software Change Manager Context Diagram

The Software Change Manager does not interact with any CSCIs or CSCs.

4.9.3.6.3.3 Software Change Manager Architecture

The Software Change Manager (ClearCase) does not interface with any external processes.

4.9.3.6.3.4 Software Change Manager Process Descriptions

The Software Change Manager's primary process is the COTS package, ClearCase. ClearCase has both a command line and a graphical user interface to execute its programs. Table 4.9.3.6.3.4-1 provides a summary of its functions.

Table 4.9.3.6.3.4-1. Software Change Manager Processes

Process	Type	COTS / Developed	Functionality
SW Change Mgr. (ClearCase)	Other	COTS with custom developed scripts	<ul style="list-style-type: none">Organizes and stores software in a software library.Manages multiple versions of software files.Regulates access to software file versions.Controls and logs changes to software file versions.Manages software file version's progress through the development cycle.Performs builds of software according to user defined version specifications.Maintains records of a build's content (files, compiler, and other resources used).

4.9.3.6.3.5 Software Change Manager Process Interface Descriptions

Software Change Manager Process Interface Descriptions Process are not applicable to the MLCI.

4.9.3.6.3.6 Software Change Manager Data Stores

The Software Change Manager's COTS package, ClearCase's data stores consist of a database and log files. Table 4.9.3.6.3.6-1 provides descriptions of the data stores shown in the Software Change Manager architecture diagram.

Table 4.9.3.6.3.6-1. Software Change Manager Data Stores

Data Store	Type	Description
ClearCase Database	Database	ClearCase uses a proprietary database management/database scheme that consists of versioned object base(s) (VOB) and views. A VOB is a data structure mounted as a multi-version file system and is created through use of the ClearCase "make vob" command. A VOB contains versions of directories and files, user-defined metadata, build records, event records, and configuration records. A view is also a data structure that's used as short-term storage for data created during the development process. View stores checked-out versions of file elements, a user's private files, and newly built derived objects.
ClearCase Log Files	Other	ClearCase log files record error and status information from various ClearCase server programs and user programs. These log files are ASCII files and are described in the ClearCase Reference Manual.

4.9.3.6.4 MLCI - Change Request Manager Computer Software Component Description

4.9.3.6.4.1 Change Request Manager Functional Overview

The Change Request Manager enables the DAACs, EOC, and SMC staffs to enter, maintain, and keep track of configuration change requests (CCRs) electronically. A COTS application called the DDTS, (Version 3.2, with some customizations) is used to perform the change request functions. Each site (System Management Center (SMC), Distributed Active Archive Centers (DAACs), and Earth Observing System Operations Center (EOC)) has a copy of the Change Request Manager. This gives the sites the capability to compose and maintain local CCRs and also compose and submit ECS CCRs to the SMC for system-wide consideration. Communications between site Change Request Managers can be established via a DDTS utility program and maintained by each site's DDTS administrator.

Description: The Change Request Manager (CRM) provides the functionality to compose, submit, coordinate, and track the status of CCRs. DDTS, a COTS application, forms the basis of the Change Request Manager and it has been customized to include a configuration change request form for inputting CCR information. DDTS records are organized into DDTS classes (note, this is not referring to object classes) and projects. Each DDTS class has its own process model and rules for how records are handled. Each DDTS class consists of one or more projects. A project is used to group DDTS records for a specific development product. The customization

of DDTS and explanations of its terminology, directories, files, and database are covered in the DDTS Administrator's Manual.

Contents: The Change Request Manager CSC contains the following custom files.

- Readme File: This file describes how to install the customized DDTS.
- Change_Request Class: The Change_Request Class was developed to handle CCR information. This is a custom DDTS class added to the ~ddts/class directory that enables the operations staff to enter configuration change request information into the DDTS database. This class was developed through the use of adminbug, a DDTS utility program that creates new classes through the use of information in the customizing section of the DDTS Administrator's Manual. Therefore, Change_Request contains the standard set of DDTS class directories and files and uses standard DDTS code formats. To implement the Change_Request class, changes were made to the following class directory elements:
 - master.tmpl: Change_Request class directory file contains the DDTS code executed when the Change_Request class is selected for use. It defines the rules for moving a CCR from state-to-state by enabling interactive dialogue and requesting the data for state transition, it defines how the fields for a CCR are displayed on the monitor or printer, and it defines how a CCR is formatted for e-mail.
 - oneofs directory : The "oneofs" directory contains a set of files where each file defines a list of valid responses for an associated field. "oneofs" files added include: CCB-Organization, Change-Class, Disposition, Eval-Organization, Impact-Evaluators, Impl-Organization, and Priority, Sites-Affected.
 - helps directory: The "helps" directory contains a set of files where each file provides descriptive information about an associated field. A help file was developed for each new field added to the database.
 - The following CCR related fields were added to the DDTS database:

baselines_affected	ccb_approval_date	ccb_approval_official
ccb_organization	change_class	ci_affected
closed_by	closing_date	closing_org
completion_date	disposition	docs_affected
effective_date	est_time_to_complete	eval_organization
impact_eval(1 - 12)	impacts_project	impl_engr
impl_engr_email	impl_organization	manager
originator_name	originator_org	originator_phone
originator_eval_engr	priority_string	related_ccr
release_affected	site_affected (1 - 9)	start_date
submitter_host	submitter_org	submitter_phone
test_est_complete_date	test_org	test_status
verify_engineer		

4.9.3.6.4.2 Change Request Manager Context Diagram

The Change Request Manager does not interact with any other CSCIs or CSCs.

4.9.3.6.4.3 Change Request Manager Architecture

The Change Request Manager (DDTS) does not interface with any other CSCIs or CSCs.

4.9.3.6.4.4 Change Request Manager Process Descriptions

The Change Request Manager's primary process is the COTS package, DDTS. DDTS has both a character based and a graphical user interface. Table 4.9.3.6.4.4-1 provides a description of the Change Request Manager COTS package.

Table 4.9.3.6.4.4-1. Change Request Manager Processes

Process	Type	COTS / Developed	Functionality
Change Request Manager (DDTS)	Other	COTS	<ul style="list-style-type: none">• Facilitates the entry, update, and recording of CCR and NCR information• Organizes and stores CCRs and NCRs in a database• Maintains the status and disposition of CCRs and NCRs as they advance through the approval and implementation processes• Provides reports of CCR and NCR information

4.9.3.6.4.5 Change Request Manager Process Interface Descriptions

Process interface descriptions are not applicable to the Change Request Manager

4.9.3.6.4.6 Change Request Manager Data Stores

The Change Request Manager's data stores consist of a database and a log file. Table 4.9.3.6.4.6-1 provides descriptions of the data stores shown in the Change Request Manager architecture diagram.

Table 4.9.3.6.4.6-1. Change Request Manager Data Stores

Data Store	Type	Functionality
DDTS Log File	Other	The DDTS log file records CCR/NCR update activity, status information and error messages from various DDTS programs.
DDTS DB	Database	DDTS has a proprietary database management/database schema and the database is described in the DDTS Administrator's Manual. The database consists of three tables: defects table (stores all of the basic CCR and NCR information), enclosures table (stores CCR/NCR related files), and change_history table (stores the complete history for each CCR and NCR as it progresses through its life cycle). Usually, there is one database per site. The DDTS procedures facilitate partial replication of a site's database at other sites.

4.9.3.6.5 MLCI - Software Distribution Manager Computer Software Component Description

4.9.3.6.5.1 Software Distribution Manager Functional Overview

The Software Distribution Manager enables the SMC and the DAAC staffs to distribute ECS software, database, software documentation, and commercial software files across a multi-platform ECS network. A COTS application called Tivoli Courier is used to perform the software distribution functions. There are no custom files.

4.9.3.6.5.2 Software Distribution Manager Context Diagram

The Software Distribution Manager, Tivoli Courier, does not interact with other system software. Tivoli Courier is installed on all of the platforms sending and/or receiving software distribution packages. Tivoli Courier on the source host platform communicates with Tivoli Courier installed on the receiving platforms.

4.9.3.6.5.3 Software Distribution Manager Architecture

The Software Distribution Manager COTS package, Tivoli Courier, is based on the Tivoli Management Platform, an architecture and set of fundamental COTS tools for managing client/server systems. Details of this architecture are provided in the Tivoli reference manuals.

4.9.3.6.5.4 Software Distribution Manager Process Descriptions

The Software Distribution Manager is a COTS product (Tivoli Courier) that handles adding, distributing, updating, and synchronizing all new software updates at a local site. Table 4.9.3.6.5.4-1 provides a description of the process shown in the Software Distribution Manager architecture diagram in the Tivoli reference manuals.

Table 4.9.3.6.5.4-1. Software Distribution Manager Processes

Process	Type	COTS / Developed	Functionality
Software Distribution Manager {Tivoli Courier}	Other	COTS	<ul style="list-style-type: none"> Provides a centralized software distribution capability to add new software, upgrade existing software with newer versions, and synchronize software on distributed systems Distributes software to multiple heterogeneous platforms concurrently Reports results of software distribution activity

4.9.3.6.5.5 Software Distribution Manager Process Interface Descriptions

Process interface descriptions are not applicable for the Software Distribution Manager.

4.9.3.6.5.6 Software Distribution Manager Data Stores

The Software Distribution Manager's data stores consist of a database and a log file. Table 4.9.3.6.5.6-1 provides descriptions of the Tivoli Courier data stores.

Table 4.9.3.6.5.6-1. Software Distribution Manager Data Stores

Data Store	Type	Functionality
Tivoli Courier Database	Database	Tivoli Courier has a proprietary database management/database schema. Its database stores records concerning file package managers which consists of a set of file packages and a list of subscribing nodes and platforms for its file packages. A file package describes the source host location, the source file paths, the receiving platform path, specific actions performed on the files when they reach their destination, and log activity.
Log File	Other	The log file contains details about the results of software distribution activity, errors, and other process messages.

4.9.3.6.6 MLCI - Software License Manager Computer Software Component Description

4.9.3.6.6.1 Software License Manager Functional Overview

The Software License Manager manages network license activities associated with using COTS products. The Software License Manager maintains information about license provisions, meters use of installed licenses, and reports on licensing events and statistics for vendor software having embedded FLEXlm licensing technology.

Software License Manager functionality is implemented by the COTS product FLEXlm and its associated data files.

The Software License Manager contains no custom scripts or files.

4.9.3.6.6.2 Software License Manager Context

Software License Manager runs at every ECS site, providing local network licensing services to requesting COTS applications and ECS operators. As shown in Figure 4.9.3.6.6.2-1, it has an interface with the Management Software CSCI, which monitor's FLEXlm's log file in order to notify the Operations Staff when "interesting" events (e.g., "license is about to expire," "no more license tokens to issue") occur. Table 4.9.3.6.6.2-1 provides a description of the interface events shown in the Software License Manager context diagram.

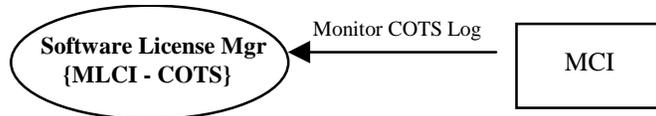


Figure 4.9.3.6.6.2-1. Software License Manager Context Diagram

Table 4.9.3.6.6.2-1. Software License Manager Interface Events

Event	Interface Event Description
Monitor COTS Log	The MCI periodically checks the COTS log files to determine if significant events have occurred. Data from identified significant events (e.g., "server died," "connection lost," or "license due to expire mm/dd/yy) are extracted and inserted into Tivoli events to be forwarded to the Operations Staff.

4.9.3.6.6.3 Software License Manager Architecture

Figure 4.9.3.6.6.3-1 is the Software License Manager architecture diagram and consists of FLEXlm and its related data files. It has a client/server architecture with license servers responding to requests from client processes embedded in managed COTS applications or license manager utilities and multiple license servers can run concurrently. It provides a command line interface for the Operations Staff.

FLEXlm consists primarily of license manager daemons, vendor daemons, license files, and client applications code embedded in licensed application. Each FLEXlm server must have its own license file, and each server logs errors and licensing events to its own "debug file". Options files are used to specify operating parameters for handling individual vendors' products. Redundant FLEXlm servers can be configured to insulate against server failure; however, this requires three license server hosts.

Tivoli monitors FLEXlm's debug logs in order to notify operators when interesting events occur.

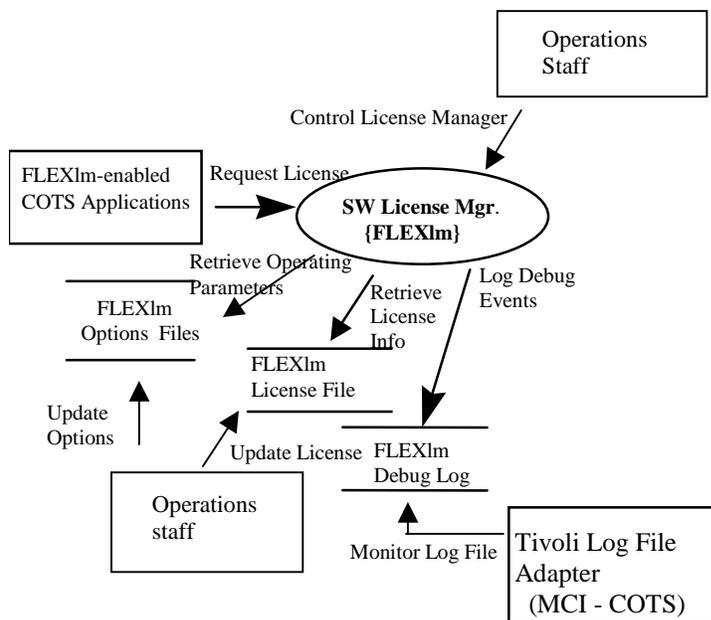


Figure 4.9.3.6.6.3-1. Software License Manager Architecture Diagram

4.9.3.6.6.4 Software License Manager Process Descriptions

License management services are performed at local sites by a set of COTS daemons and utilities identified collectively in Table 4.9.3.6.6.4 as the Software License Mgr. The principal persistent process is the server daemon, `lmgrd`. It controls interactions between client processes and vendor daemons. The latter grants or denies an application’s request to use a license. Table 4.9.3.6.6.4-1 provides a description of the process involved in software licenses management for a local site.

Table 4.9.3.6.6.4-1. Software License Manager Processes

Process	Type	COTS / Developed	Functionality
Software License Mgr. (FLEXlm)	Other	COTS	<p>The FLEXlm server daemon (lmgrd) with its associated command line utilities:</p> <ul style="list-style-type: none"> • shuts down and restarts license daemons on a license server node and makes license data available to the servers • manages license checkout and checkin processing for FLEXlm-enabled COTS products • logs licensing events and errors to files on the local network • removes a user's license for a specified feature • displays the status of installed licenses and of network licensing activities; this includes listing licensed software features and their associated product versions, vendors, hosts, and expiration dates • reports the hostid of a system (needed to obtain license key from vendors)
FLEXlm-enabled COTS Application	Other	COTS	Client software within vendor products communicates with FLEXlm's license server and vendor daemons to request licenses for product users to run.

4.9.3.6.6.5 Software License Manager Process Interface Descriptions

Most of the Software License Manager's process interfaces exist between the license servers and their client applications embedded within COTS applications to handle license requests. They are not described here in detail, as they are internal to the COTS software.

Table 4.9.3.6.6.5-1 provides descriptions of the interface events shown in the Software License Manager architecture diagram.

Table 4.9.3.6.6.5-1. Software License Manager Process Interface Events

Event	Event Frequency	Interface	Initiated By	Event Description
Control License Manager	One per event	Process: SW License Mgr. (COTS) (FLEXIm and iFOR/LS)	Operations Staff	The Operations Staff start, stop, and monitor license servers, as well as update license manager databases, and request reports about installed licenses and selectable license events.
Log Debug Events	One per debug event	<i>Data Store:</i> FLEXIm Debug Log	<i>Process:</i> SW License Mgr. (FLEXIm) (COTS)	The FLEXIm server logs all errors and license events (e.g., start/stop date, and status of installed license, product version, host(s) to which servers are connected).
Monitor Log File	One per event	<i>Data Stores:</i> FLEXIm Debug Log, iFOR Log File	<i>Process:</i> Tivoli Log File Adapter (COTS)	Reads and extracts messages logged since the last read event.
Update License	One per update request	<i>Data Store:</i> FLEXIm License File	Operations Staff	The Operations staff adds and removes the managed FLEXIm software licenses.
Retrieve License Info	One per request	<i>Data Store:</i> FLEXIm License File	<i>Process:</i> SW License Mgr. (FLEXIm) (COTS)	Reads license provisions for one or more FLEXIm-enabled COTS products.
Update Options	One per parameter change	<i>Data Store:</i> FLEXIm Options Files	Operations Staff <i>Process:</i> Unix file editor	Store parameters for regulating license usage and event logging (FLEXIm).
Retrieve Operating Parameters	One per license request	SW License Mgr.(COTS) (FLEXIm/iFOR/LS)	<i>Process:</i> SW License Mgr. (FLEXIm) (COTS)	Retrieve parameters for regulating license usage and event logging.
Request License	One per license request	<i>Process:</i> SW License Mgr. (COTS) (FLEXIm and iFOR/LS)	<i>Programs:</i> FlexIm- and iFOR/LS-enabled COTS applications	Communication among license servers and clients to establish connections and checkout, check-in, and monitor activity of licenses.

4.9.3.6.6.6 Software License Manager Data Stores

License Manager's principal data stores are the FLEXIm license, debug, and option files; the iFOR/LS database, nodelock, and user files, and the iFOR day log file. FLEXIm files are described in the FLEXIm End User Manual. iFOR/LS files are described in the iFOR/LS

Administrator's Guide. Table 4.9.3.6.6-1 provides descriptions of the data stores shown in the Software License Manager architecture diagram.

Table 4.9.3.6.6-1. Software License Manager Data Stores)

Data Store	Type	Functionality
FLEXIm license file	text file	<p>A collection of records containing license provisions and passwords for one or more FLEXIm-enabled COTS products. They identify:</p> <ul style="list-style-type: none"> • servers - name, hostid, and port number of the license manager daemon • daemons - name and path of vendor daemons that track licenses checked out and to whom. An options file can be named for each vendor daemon • features - description of the license to use a product <p>Each license server uses one license file, and operators combine license files received from vendors, as possible, to reduce the number of servers in the network. License file content and format is described in the FLEXIm End User Manual.</p>
FLEXIm debug log	text file	<p>A collection of records describing licensing errors and events that have occurred. Records contain a timestamp, an informational message, and the name of the daemon generating the message. Each license server (except redundant servers) writes to its own log file.</p>
FLEXIm options files	text file	<p>Collections of records that specify optional operating parameters for managing specific vendors' products. Options files are named in license files. There can be one options file for each vendor each license file specifies.</p>

4.9.4 ECS Assistant Script Library

4.9.4.1 ECS Assistant Script Library Functional Overview

ECS Assistant Script Library supports the ECS Assistant GUI and almost all-internal ECS shell scripts for system installation, configuration, monitoring and shutting down servers and many other functions. These functions are described in the Process Interface Descriptions sub-section.

4.9.4.2 ECS Assistant Script Library Context

The context diagram is not applicable to the ECS Assistant Script Library.

4.9.4.3 ECS Assistant Script Library Architecture

The architecture diagram is not applicable to the ECS Assistant Script Library.

4.9.4.4 ECS Assistant Script Library Processes

The Process Table is not applicable to the ECS Assistant Script Library.

4.9.4.5 ECS Assistant Script Library Process Interface Descriptions

The ECS Assistant Script Library provides functions available to calling scripts. Table 4.9.4.5-1 describes the interface.

Usage: /ecs/formal/COMMON/scripts/EcCoScriptlib command [args]

Table 4.9.4.5-1. ECS Assistant Script Functions (1 of 3)

Function	Functionality
add_cds_entry	Add CDS entry.
add_client	Add DCE client.
archive_if_needed	Save a copy of the destination file if different from the original file.
cache_subsys_components	Cache the subsystem-component information.
cfgpatch	Patch parameter files based on the .cfgpatch file.
check_dirs	Check for the existence of specified directories.
clean_logs_dir	Remove logs in the log directory older than a specified date.
cleanup	Cleanup after servers in a subsystem and component.
components_health	Show health of component installation.
convert_shell	Convert a file to a format to be included by the specified shell.
date_to_daynum	Compute the date from the day number in the year.
daynum_to_date	Compute the day number in the year from the date.
days_ago	Compute the date num_days ago.
del_cds_object	Delete server object from CDS name space.
extract_desc_value	Extract a value from an ESDT descriptor file, based on its object name.
generate_servers_file	Generate servers file for a given mode, subsystem and component.
get_architecture	Display the ARCH variable that identifies the machine.
get_colorpair	Get color pairs from a predetermined palette of color pairs.
get_health	Display installation health indicators for mode, subsystem and component.
get_hostlist	Get the host list for the current site.
get_hostname	Display the HOSTNAME variable that identifies the machine.
get_included_files	Get names of included files from a package.
get_revlevel	Display the revision level of a parameter file.
get_site	Get the site we are running on.
get_site_acronym	Get the standard three-letter acronym of the site we are running on.
get_taglist	Get a list of tags from a file.
getuname	Print user name.
getview	Print Clearcase view or NONE.
install	Install to a mode.
install_type	Determine the installation type of an installation.
is_leap_year	Determine whether the year is a leap year.
kill	Kill servers in a subsystem and component.
kill_procs	Kill the named processes.

Table 4.9.4.5-1. ECS Assistant Script Functions (2 of 3)

Function	Functionality
list_apps	List applications in a mode, subsystem and component.
list_apps_by_ius	List applications by which IUs are installed.
list_archpkgs	List packages for a particular architecture.
list_components	List components of a subsystem.
list_componentsdirs	List components' directories of subsystem.
list_execs	List executables of a given type in a mode, subsystem and component.
list_execs_by_app	List executables, which belong to the specified application.
list_execs_by_ius	List executables of a given type in a mode by which IUs are installed.
list_installtypes	List file types, which can be installed.
list_ius	List installed units in a mode, subsystem and component.
list_logfiles	List the log files for the servers in a subsystem and component.
list_mkcfgs	List the Mkcfcg scripts, which are available for execution.
list_servers	List servers in a subsystem and component.
list_servers_from_file	List servers in a .servers file.
list_subsystems	List subsystems.
look_for_proc	Look for a process.
lookup_component	Lookup details on a component.
lookup_installtype	Look up a file type, which can be installed.
lookup_subsystem	Lookup details on a subsystem.
mk_cds_entry	Make CDS entries for a subsystem and component.
mkcfg	Make config files for servers in a subsystem and component.
mkmodedirs	Make all the directories required for a mode.
mode_health	Display health indicators for a mode.
modify_parms	Modify a dbparms, extparms or cfgparms file.
monitor	Monitor servers in a subsystem and component - continuously.
monitor_once	Monitor listed servers - a one-time snapshot.
package_health	Show health of a package installation.
record_proc	Record a process.
refresh_common_scripts	Refresh common ECS scripts.
refresh_control_files	Refresh ECS control files.
revert_parms	Recover the most recently time stamped parameters file.
set_cfgparms	Set configurable parameters in the environment.
set_dbparms	Set database parameters in the environment.
set_environment	Write the appropriate commands to set up the standard environment.
set_fileperms	Set file permissions.
set_symbolic_hosts	Set up the array of symbolic host names for this site.
start	Start servers in a subsystem and component.
start_client	Start an ECS client program.
start_ecs	Start the whole ECS system.

Table 4.9.4.5-1. ECS Assistant Script Functions (3 of 3)

Function	Functionality
start_gui	Start an ECS GUI.
start_server	Start a single ECS server with standard checks.
start_server_group	Start a group of servers with standard checks.
strip_comments	Strip comments and blank lines from a file.
subsys_health	Show health of subsystem installation.
Uninstall	Uninstall from a mode.
Viewlog	View logs.

4.9.4.6 ECS Assistant Script Library Data Stores

ECS Assistant data files have some common properties. All files support comments beginning with a "#" sign and terminating at the end of the line. Also, blank lines are permitted. All blank lines and comments are ignored in the processing of the file.

Applications File (.applications)

The .applications file is delivered from /ecs/formal/COMMON/.applications in Clearcase to \$ECS_HOME/\$MODE/.applications in the mode. It is always delivered whenever ECS Assistant is used to install.

The format of this file is shown in Table 4.9.4.6-1. The file maps application names to application ids. These application ids are used by HP Open View to track the ECS applications during operations.

Table 4.9.4.6-1. Applications File Format

Header	
NAME	Application ID
EcEcEcsApp (System - The Boss Application)	9999999
CSS	
EcCsLandsat7GatewayApp	6000100
EcSbSubServerApp	6000000
EcCsMojoGatewayApp	6000200
EcGwAsterGatewayApp	6000300

Cfgparms File (.cfgparms)

The .cfgparms files have a header, several tags denoted by a word ending in a colon ":", and one or many parameter value pairs associated with each tag. There is one .cfgparms file associated with each subsystem - component pair. For each Mkcfg script which exists for a component (whether it is delivered or not), there is a tag and the parameter value pairs which describe the configurable parameters for a particular subsystem and component. It is always delivered

whenever ECS Assistant is used to install that particular subsystem and component. Table 4.9.4.6-2 is the .cfgparms file format.

Table 4.9.4.6-2. Cfgparms File Format

Header	
EcDsScienceDataServerAppMOscfg:	
NumOfHDFServer	3
EcDsHdfEosServerMkcfg:	
AppLogSize	1000000
AppLogLevel	0
DebugLevel	0
DBLibrary	SYBASE_CT
DBServer	OTIS_SERVER

Cfgpatch File (.cfgpatch)

The .cfgpatch file specifies all the changes between one drop of code and another, between releases. The parameter REVISION_LEVEL needs to be defined in the file so the ECS Assistant software can check whether a .cfgparms file needs to be patched or not. The syntax for having the REVISION_LEVEL parameter expanded before execution of the patching is %\${REVISION_LEVEL}. The first few lines in this cfgpatch file are standardized, and they add the parameter RevisionLevel to the .cfgparms file with the value specified in the REVISION_LEVEL definition:

```
DEFINE REVISION_LEVEL 5A.05.EPSILON.01
```

Subsequent lines in the file specify parameters and tags to add and delete in cfgparms and dbparms files. The possible *actions* are: *DEFINE*, *ADD*, *UPD*, *DEL*, *ADD_TAG*, *DEL_TAG*. The *DEFINE* action takes two additional arguments, *Parameter* and *Value*. The *ADD* and *UPD* actions need all 10 fields specified. The *DEL* action does not need *valuelist* specified. The *DEL_TAG* action does not need *Parameter* or *ValueList* specified. *Site* can be any of the supported sites listed in the sitemap or *ALL*. *Host* is any host name at the specified site or *ALL*. *Mode* is any supported mode or *ALL*. *Subsys* is any subsystem label as specified in the subsystems file. *Component* is one of the components specified in the components file for that subsystem. *Filename* can be either .cfgparms, .dbparms or .extparms. *Tag* can be any tag in the .cfgparms file, or could be a new tag to add. *Parameter* can be any string without spaces, and starting with an alphanumeric character. *ValueList* can be any string, even with spaces in it.

A sample file is shown in table 4.9.4.6-3.

Table 4.9.4.6-3. Cfgpatch file format

Define a macro										
Action	Parameter	Value								
DEFINE	REVISION_LEVEL	5a.05.EPSILON.01								
ALL files										
Action	Site	Host	Mode	Subsys	Comp	Filename	Tag	Parameter	ValueList	
ADD	ALL	ALL	ALL	ALL	ALL	.dbparms	Rev Lev el	Revision Level	%\${REVIS ION_LEV EL}	
ADD	ALL	ALL	ALL	ALL	ALL	.cfgparms	Rev Lev el	Revision Level	%\${REVIS ION_LEV EL}	
GSFC .cfgparms changes										
Action	Site	Host	Mode	Subsys	Comp	Filename	Tag	Parameter	ValueList	
ADD	GSFC	ALL	ALL	CLS	EcCl	.cfgparms	EcC IWb Jdt Mkc fg	SUBSC RIPTIO N_ESD T_SN	AST_L1B T	
ADD	GSFC	ALL	ALL	CLS	EcCl	.cfgparms	EcC IWb Jdt Mkc fg	SUBSC RIPTIO N_ESD T_VID	001	

Components Files (.components)

The components file specifies which logical components exist under a subsystem. For example, the DSS subsystem has the following components: EcDsSr, EcDsSt, EcDsDd, and EcDsDo. They are specified in the components file for that subsystem. Each subsystem will have a components file, even if there is only one component. For example, Ingest has only one component, EcIn. The components file specifies the component names and the directory under the main subsystem directory where the ECS Assistant files for that component can be found. Table 4.9.4.6-4 is a sample of the file format.

Table 4.9.4.6-4. Components file format

Mnemonic	Directory	CSCI
EcDsSr	SDSRV	Science Data Server
EcDsSt	STMGT	Storage Management
EcDsDd	Distribution	Data Distribution
EcDsDo	DDSRV	Document Data Server

Dbparms File (.dbparms)

The dbparms file has the exact same format as the .cfgparms file, except that the tags refer to database scripts, and the parameter value pairs apply to values used in the database operations. There is one dbparms file delivered for each subsystem component pair, whether there is a database or not. If there is no database, the file will have only a header and no tags or parameter value pairs.

Installable Unit (IU) and Files

An installable unit file contains several types of entries. There is usually a header consisting of comments. The file may have one or more tags denoted by a word ending in a colon. Those tags come from the installtypes file, see below. There are also some pseudo tags. The tags *preinstall*, *postinstall* are used to invoke an action before or after the main installation. ANY UNIX COMMAND can be put there. There is also the *ssunique* tag, which allows for installation of objects not referenced by the installtypes file. Associated with the *ssunique* tag, you can have commands of the form *copy*, *link*, *cpln*, and *mkdir*. The *copy* command takes two arguments, *orig* specifying the file to be copied, and *dest* specifying the location to copy the file to. Macros specified in the EA macros file are used when specifying these paths, so that hard coding of paths can be eliminated. The *link* command can be used to make a symbolic link from one file or directory to another. The *cpln* command will make a symbolic link in a development installation and a copy in a non-development installation. The *mkdir* command will make the specified directories if they do not exist, and will not complain if they exist already. The usage of the four commands is as follows:

```
copy [-r] permissions orig dest
      link orig dest
      cpln permissions orig dest
      mkdir dir dir ...
```

An IU file may also include another file using the `%include` command. It takes one file name as an argument. It is generally only used in PKG files, and not in IU files.

Eamacros File (.eamacros)

This file lists the standard ECS Assistant macros available to be used in IU and PKG files.

Envvars File (.envvars)

This file occurs in the /ecs/formal/COMMON and in every subsystem. It specifies in an architecture specific way the environment to be used when running servers. The file is converted to a shell script file, which is then read in at run time to create the standard environment. It consists of several tags: *allhosts* which is applied to all architectures, *sun5.5*, which applies to SUN hosts, *hp10*, which applies to HP hosts, *sgi6n32*, which applies to SGI hosts. The other entries are in the form of parameter value pairs. The resulting file will consist of entries of the form

```
export parameter="{value}"
```

These commands are executed at run time and read into the environment in a Korn shell script. This same format applies to the common and subsystem-component levels.

Executables File (.executables)

The executables file specifies all the executable servers, GUIs, cgi-bin programs, clients and utilities. The file has a header and five fields for each un-commented line as follows:

Program is the executable name

IU Name is the IU file, which delivers the program

Type, can be S for server, C for client, G for Gui, T for test program, U for utility, or W for CGI Web interface

Program Id is a number assigned to that program by the Architect's Office

Application name is a name associated with the application the executable belongs to, and which must be in the applications file.

Table 4.9.4.6-5 is a sample of the file format.

Table 4.9.4.6-5. Executables file format

Program	IU Name	Type	ProgID	Application Name
EcCIodProductRequest	EcCIodAsterOnDemand.iu	C	1000005	EcCIodProductRequestApp
EcCIDtDesktopDaacUser	EcCIDtDesktopDaacUser.iu	G	1000100	EcCIDtDesktopApp
EcCIDtDesktopSciUser	EcCIDtDesktopSciUser.iu	G	NONE	EcCIDtDesktopApp
NOTES:				
Program Types				
S- Server	C – Client	G – Gui	T – Test Program	U – Utility
W – CGI Web Interface				

Hostmap File (.hostmap)

The hostmap file has a header and uncommented entries of the form *site : host*. This file maps the hosts at each site to the site name. Table 4.9.4.6-6 is a sample of the file format.

Table 4.9.4.6-6. Hostmap file format

Site : Host
EDC:e0acg01
EDC:e0acg02
EDC:e0acg03
EDC:e0acg04
EDC:e0acg05
EDC:e0acg06

Installtypes Files (.installtypes)

The installtypes file format is described in table 4.9.4.6-7. It consists of a file type tag, the permissions to be used for the installation of that type, three location fields describing where the information comes from, and one describing where it goes in the mode. The fields use the standardized macros in the eamacros file.

Table 4.9.4.6-7. Installtypes file format

Installtype	Description
[File Type]	Type of file to install in square brackets
Permissions	Permissions to be set on installation
Orig MODE staging Copy	Location to copy from for MODE staging
Orig CM_BUILD Copy	Location to copy from for CM_BUILD
Orig NORMAL or STAGE Copy	Location to copy from for Normal or STAGE
Dest. Copy	Location in mode to copy to
Binary Files [BIN] 755	
	<code>\${MODE_STAGING_DIR}/CUSTOM/bin/\${SUBSYS_DIR_MODE}</code>
	<code>\${CMTOP}/\${SUBSYS_DIR_CM}/install/CUSTOM/bin/\${SUBSYS_DIR_MODE}</code>
	<code>\${CMTOP}/\${STAGE}/\${SUBSYS_DIR_CM}/bin/\${ARCH}</code>
	<code>\${BINDIR}</code>

Package (PKG) Files

A package file supports exactly the same format as an IU file, but is generally used only to include IU files with the %include command. This allows the system designers to install a set of IU files as one package.

Packages File (.packages)

The packages file specifies which packages are delivered for a particular architecture. The package name, subsystem, and component identify the package. The platforms are specified with one or more of the tags *HP*, *SGI*, *SUN* separated by pipe symbols "|". The packages file is used by the EcCoMkDeliver script to determine which packages need to be delivered when making the tar files for a drop on a particular architecture. Table 4.9.4.6-8 is a sample of the file format.

Table 4.9.4.6-8. Packages file format (1 of 2)

Package	Subsys	Comp	Platforms	Comment
.EcCIEOSView.pkg	CLS	EcCI	SUN	
.EcCIINTFCSVR.pkg	CLS	EcCI	SUN	
.EcCIINTFCSVR_EDC.pkg	CLS	EcCI	SUN	
.EcCIJdt.pkg	CLS	EcCI	SUN	
.EcCIOdAsterOnDemand.pkg	CLS	EcCI	SUN	
EcCsCommon.pkg	CSS	EcCs	HP SGI SUN	
EcCsDarMainMSSSVR_EDC.pkg	CSS	EcCs	HP	
EcCsINTFCSVR.pkg	CSS	EcCs	Sun	
EcCsINTFCSVR_EDC.pkg	CSS	EcCs	Sun	
EcCsINTFCSVR_GSFC.pkg	CSS	EcCs	Sun	
EcCsMojoGateway.pkg	CSS	EcCs	Sun	
EcCsSubServerGUI.pkg	CSS	EcCs	Sun	
EcCsRegistryGUI.pkg	CSS	EcCs	Sun	

Sitehostmap File (.sitehostmap)

The sitehostmap file specifies the mapping of symbolic hosts to actual hosts at the various DAACs. The format is to use one column per site with the first column indicating the symbolic host name. The format is getting to be unwieldy as we add new DAACs. Putting the whole thing into a database will solve this. The format is shown in table 4.9.4.6-9.

Table 4.9.4.6-9. Sitehostmap file format (1 of 2)

Symbolic Name	E D C	G S F C	L a R C	M D A A C	M S M C	N S I D C	S M C	V A T C	V S M C	E D F F R T
AsterDemWs	e 0 a s s 0 3									
AsterLutDb01	e 0 a s s 0 1									
AsterLutDb02	e 0 a s s 0 2									
BbsServer					u h o h		m 0 c s s 0 2		t 1 s m s 1 2	
CssServer	e 0 c c s s 0 2	g 0 c c s s 0 2	1 0 c c s s 0 2	o n f i r e	c h a m m y	n 0 c c s s 0 2	m 0 c c s s 0 3	t 1 c c s s 0 1	t 1 s m s 0 9	

Table 4.9.4.6-9. Sitehostmap file format (2 of 2)

Symbolic Name	E D C	G S F C	L a R C	M D A A C	M S M C	N S I D C	S M C	V A T C	V S M C	E D F F R T
FtpServer01					m o n d a y		m 0 c s s 0 5		t 1 s m s 1 1	
FtpServer02							m 0 c s s 0 4			
DbasOperationWs	e 0 d d m h 0 0 2	g 0 d d m h 0 0 1	1 0 d d m h 0 0 2	d a r k w i n d		n 0 d d m h 0 2		t 1 d d m h 0 3		

Sitemap File (.sitemap)

The sitemap file specifies the packages that can be installed on particular hosts at particular sites by ECS Assistant. The entries consist of a 4-tuple consisting of site : host : subsystem : component followed by a list of packages, one to a line. The format is shown in table 4.9.4.6-10.

Table 4.9.4.6-10. Sitemap file format (1 of 2)

Entry	Package(s)
EDC:e0ais02:CLS:EcCl # AIT Workstation (Default)	EcCIEOSView.pkg
EDC:e0ais03:CLS:EcCl # AIT Workstation/DBMS Server	EcCIEOSView.pkg

Table 4.9.4.6-10. Sitemap file format (2 of 2)

Entry	Package(s)
EDC:e0ins01:CLS:EcCI # Interface Server 02 # Principal for CSS, Backup for DMS, IOS, CLS Servers	.EcCsINTFCSVR.pkg .EcCsINTFCSVR_EDC.pkg .EcCIJdt.pkg .EcCIOdAsterOnDemand.pkg
EDC:e0ins02:CLS:EcCI # Interface Server 01 # Principal for DMS, IOS, CLS Servers, Backup for CSS	.EcCsINTFCSVR.pkg .EcCsINTFCSVR_EDC.pkg .EcCIJdt.pkg .EcCIOdAsterOnDemand.pkg
EDC:e0pls03:CLS:EcCI # Planning/Management WS 01 (Default)	EcCIEOSView.pkg

Subsystems File (.subsystems)

The subsystems file consists of four fields. The four fields are:

- a *two letter acronym* (obsolete)
- a *project acronym*, which is used as the principal identifier of a subsystem in all ECS Assistant code
- a *clearcase location*, which specifies the location of the ECS Assistant files in clearcase relative to /ecs/formal
- a *mode location*, which specifies the name of the directory used in the modes relative to the COMMON/bin, COMMON/lib, COMMON/www, directories)

There are other directories, which have subsystem sub-directories. The format is shown in table 4.9.4.6-11.

Table 4.9.4.6-11. Subsystems file format (1 of 2)

Two Letter Acronym	Project Acronym	Clearcase Location	Mode Location	Description
cl	CLS	CLS	CLS	Client
cs	CSS	CSS	CSS	Communications
dm	DM	DM	DMS	Data Management
dp	DPS	PDPS/DPS	DPS	Data Processing
ds	DSS	DSS	DSS	Data Server
es	ESDT	ESDT	ESDT	ESDT
in	INGEST	INGEST	INS	Ingest
io	IOS	IOS	IOS	Interoperability

Table 4.9.4.6-11. Subsystems file format (2 of 2)

Two Letter Acronym	Project Acronym	Clearcase Location	Mode Location	Description
ms	MSS	MSS	MSS	Management
pl	PLS	PDPS/PLS	PLS	Planning
tk	TOOLKIT	TOOLKIT	TOOLKIT	Toolkit
v0	VOC	V0_Client	V0_Client	V0 Client

ECS Assistant GUI

The ECS Assistant GUI calls the ECS Assistant Script Library for what it needs to interface with the underlying installation system. It calls the ECS Assistant Script Library for the functions shown in Table 4.9.4.6-12.

Table 4.9.4.6-12. Functions

Function	Description
Cfgpatch	Patch parameter files based on the .cfgpatch file.
check_dirs	Check for the existence of specified directories.
clean_logs_dir	Remove logs in the log directory older than a specified date.
get_health	Display installation health indicators for mode, subsystem and component.
get_included_files	Get names of included files from a package.
get_revlevel	Display the revision level of a parameter file.
install	Install to a mode.
list_apps	List applications in a mode, subsystem and component.
list_components	List components of a subsystem.
list_execs_by_app	List executables, which belong to the specified application.
list_mkcfgs	List the Mkcfg scripts, which are available for execution.
list_servers	List servers in a subsystem and component.
list_servers_from_file	List servers in a .servers file.
lookup_component	Lookup details on a component.
mk_cds_entry	Make CDS entries for a subsystem and component.
revert_parms	Recover the most recently time stamped parameters file.

ECS Start Scripts

The ECS Start scripts call the ECS Assistant Script Library for a variety of services, which are shown in table 4.9.4.6-13.

Table 4.9.4.6-13. Start Scripts

Start Script	Functionality
check_dirs	Check for the existence of specified directories.
get_architecture	Display the ARCH variable that identifies the machine.
getuname	Print user name.
getview	Print Clearcase view or NONE.
install_type	Determine the installation type of an installation.
look_for_proc	Look for a process.
record_proc	Record a process.
set_cfgparms	Set configurable parameters in the environment.
set_environment	Write the appropriate commands to set up the standard environment.
start_client	Start an ECS client program.
start_ecs	Start the whole ECS system.
start_gui	Start an ECS GUI.
start_server	Start a single ECS server with standard checks.
start_server_group	Start a group of servers with standard checks.
strip_comments	Strip comments and blank lines from a file.

ECS Mkcfig Scripts

The ECS Mkcfig scripts call the ECS Assistant Script Library for a variety of services, which are shown in the table 4.9.4.6-14.

Table 4.9.4.6-14. Mkcfig Scripts

Mkcfig Script	Functionality
check_dirs	Check for the existence of specified directories.
get_architecture	Display the ARCH variable that identifies the machine.
get_hostname	Display the HOSTNAME variable that identifies the machine.
get_site_acronym	Get the standard three-letter acronym of the site we are running on.
getuname	Print user name.
set_cfgparms	Set configurable parameters in the environment.
set_environment	Write the appropriate commands to set up the standard environment.
set_fileperms	Set file permissions.
strip_comments	Strip comments and blank lines from a file.

4.9.5 Systems Management Subsystem Hardware Components

4.9.5.1 MHCI Description

The MSS-MHCI include the following: two Application Servers, one MSS File Server, one CM (configuration management) server, two MSS Servers, one Tape Backup Server, and multiple PCs.

The Application Servers are SUN Server class machines. Detail specifications can be found per the site-specific, hardware design diagram, baseline document number 920-TDx-001. Because of their common configuration, these hosts can be configured interchangeably. Two MSS software CSCIs, MCI and MACI run on these hosts. Some of the key MCI functions are the MSS database management system and accountability management. As part of the MACI, custom and SNMP agents are configured to monitor and/or control managed objects distributed across heterogeneous platforms. Detailed mappings can be found per the site-specific hardware/software mapping, baseline document number, 920-TDx-002 (Revision 13 for Release 5B).

A SUN SPARC Storage Array is dual ported between both hosts and provides storage for the MSS database management system and the IQ Report Writer tool. Detail configuration is specified per common disk partition, baseline document number 912-TDx-002.

The MSS File Server and CM Server are SUN Workstation class machines. Detail specifications can be found per the site-specific, hardware design diagram, baseline document number 920-TDx-001. Both servers are configured similarly with additional RAM allocated to the File Server due to file distribution loading. Two MSS software CSCIs, MLCI and MACI, run on these hosts. Some of the key MLCI functions are the Baseline Manager (XRP), Software Change Manager (ClearCase) and Change Request Manager (DDTS). As part of the MACI, custom and SNMP agents are configured to monitor and/or control managed objects distributed across heterogeneous platforms. Detailed mappings can be found per the site-specific hardware/software mapping, baseline document number 920-TDx-002. Additional functionality provided by the File Server includes storage and processing of home directories, automounted COTS and distribution of custom code.

A SUN SPARC Storage Array is dual ported between both hosts and provides storage for the ClearCase VOBs and Views, DDTS, XRP, home directories, automounted COTS and distribution space. Detail configuration is specified per site specific disk partition, baseline document number 922-TDx-011.

The MSS Server and MSS Server Backup are HP High End Workstation class machines. Detail specifications can be found per the site-specific, hardware design diagram, baseline document number 920-TDx-001. Two MSS software CSCIs, MCI and MACI, run on these hosts. Some of the key MCI functions are network, enterprise, fault and performance management. These are all supported by a combination of Tivoli and HP OpenView COTS products. An additional key MCI component is trouble ticket (Remedy). As part of the MACI, custom and SNMP agents are configured to monitor and/or control managed objects distributed across heterogeneous platforms. Detailed mappings can be found per the site-specific hardware/software mapping, baseline document number 920-TDx-002.

A HP RAID device is dual ported between both hosts and provides storage for Remedy, HP OpenView, and Tivoli data. Detail configuration is specified per site specific disk partition, baseline document number 922-TDx-031.

The Tape Backup Server is a SUN Server class machine. Detail specifications can be found per the site-specific, hardware design diagram, baseline document number 920-TDx-001. This is a standalone host, which serves as the front end to a DLT (Digital Linear Tape Library) used for global system DAAC backups. Two MSS software components run on this host and are the MCI and MACI. The key MCI functions are the network backup and restore components (Legatto Networker). As part of the MACI, Tivoli clients and SNMP agents are configured to monitor and/or control managed objects distributed across heterogeneous platforms. Detailed mappings can be found per the site-specific hardware/software mapping, baseline document number 920-TDx-002.

A DLT with more than 1 TB of capacity is directly attached to the Tape Backup Server. Via the Legatto Networker Server, system data copies and restores are performed with the Tape Backup Server functioning as the intermediate between the Legatto clients and the DLT. Detail configuration is specified per site specific disk partition, baseline document number 922-TDx-031.

Multiple Pentium PCs are used at each DAAC site in support of office automation requirements. Detail specifications can be found per the site-specific, hardware design diagram, baseline document number 920-TDx-001. These are standalone hosts, which enable operators to perform policy and procedure management. One MSS software component runs on this host and is the MCI. Detailed mappings can be found per the site-specific hardware/software mapping, baseline document number 920-TDx-002.

In general, custom code and applications are loaded on the internal disks of all hosts. This prevents dependencies on specific hosts or any peripherals. For cost efficiency, selective application servers are stored in a RAID and accessed by one host at any time.

4.10 Internetworking Subsystem (ISS) Overview

The Internetworking Subsystem (ISS) contains one hardware configuration item (HWCI), the Internetworking HWCI. INCI provides internetworking services based on protocols and standards corresponding to the lower four layers of the OSI reference model as described below.

Transport Protocols

ECS provides IP-based connection-oriented and connectionless transport services. The connection-oriented service is implemented using TCP, while User Datagram Protocol (UDP) is used for connectionless transport. Higher layer applications use one or the other based on such requirements as performance and reliability.

Transmission Control Protocol (TCP), specified in RFC 793 (as of 08//98), is a connection-oriented, end-to-end reliable protocol designed to fit into a layered hierarchy of protocols to support multi-network applications. It provides for reliable inter-process communication between pairs of processes in host computers attached to networks within and outside ECS. Because TCP assumes it may obtain potentially unreliable datagram service from the lower level protocols, it involves additional overhead due to the implementation of re-transmission and acknowledgment processes.

The UDP, specified in RFC 768 (as of 08//98), provides a procedure for application programs to send messages to other programs with minimal overhead. The protocol is transaction oriented and delivery of data is not guaranteed, since there is no acknowledgment process or re-transmission mechanism. Therefore, applications requiring ordered and reliable delivery of data would use TCP.

Network Layer Protocols

The network layer provides the functional and procedural means to transparently exchange network data units between transport entities over network connections, both for connection-mode and connectionless-mode communications. It relieves the transport layer from concern of all routing and relay operations associated with network connections.

The Internet protocol (IP) Version 4, specified in RFC 791 (as of 08//98), is the ECS supported network protocol, based on its dominance in industry usage and wide community support. As part of IP support, ICMP and ARP are also supported.

Physical/Datalink Protocols

Physical and data-link protocols describe the procedural and functional means of accessing a particular network topology. For the DAAC and SMC networks, the data-link/physical protocols are Fiber Distributed Data Interface (FDDI) and Ethernet. (FDDI is a 100Mbps token-passing network topology, and Ethernet is a 10/100 Mbps bus topology.)

High-Performance Parallel Interface (HIPPI) networks form part of the networks at some DAACs (GSFC, LaRC, and EDC) to handle the high data volumes between the Processing and

Data Server subsystems. The HIPPI implementation involves running IP-over-HIPPI. (Large TCP window sizes are used in order to achieve high throughput rates on the HIPPI networks.)

Other technologies such as Gigabit Ethernet and Asynchronous Transfer Mode (ATM) are being considered for insertion in ECS DAAC networks.

Internetworking Hardware HWCI (INCI)

This HWCI provides the networking hardware for internal and external DAAC, SMC, and EOC connectivity. The HWCI includes FDDI switches, concentrators and cabling; Ethernet hubs and cabling; routers and cabling; HIPPI switches and cabling; and network test equipment. Each network hardware device is discussed in detail in Section 4.10.2

4.10.1 Internetworking Subsystem Description

4.10.1.1 DAAC LAN Architecture

This section provides an overview of the DAAC network architecture. Information on DAAC specific implementation level detailed designs can be found in Section 4.10.1.5.

The generic architecture for DAAC Local Area Networks (LANs) is illustrated in Figure 4.10.1.1-1. The topology consists of a User Network (FDDI at all sites), a Production Network (FDDI at all sites), and a HIPPI Network (for processing to data server flows at GSFC, EDC, and LaRC). The creation of separate User and Processing networks allows processing flows to be unaffected by user pull demands, and the introduction of the high-speed HIPPI Network provides adequate bandwidth to the Processing and Data Server subsystems' need to transfer large volumes of data. Each of the networks is discussed in more detail below.

Note that not all sites have the complete complement of hardware and subsystems shown in Figure 4.10.1.1-1. For instance NSIDC does not have a HIPPI network because HIPPI is not needed to satisfy the relatively moderate processing flows.

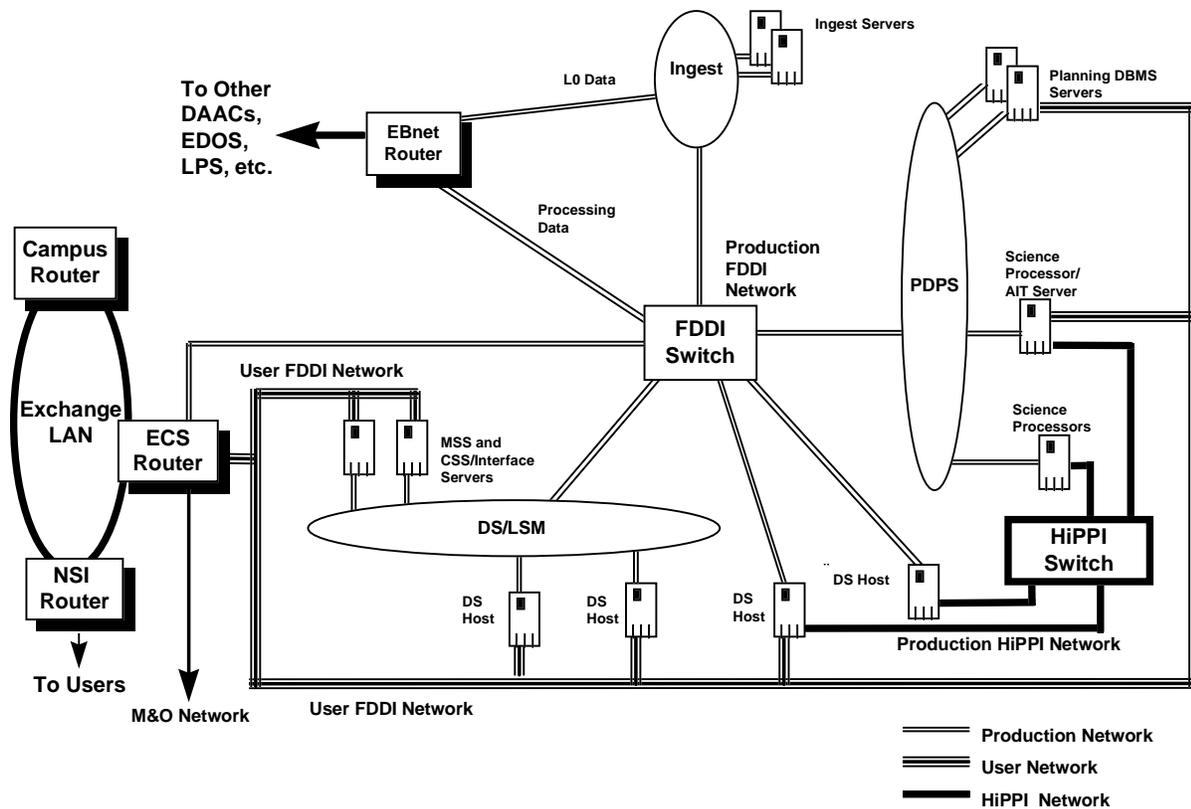


Figure 4.10.1.1-1. DAAC Networks: Generic Architecture Diagram

The Production Network consists of multiple FDDI rings supporting the DAAC subsystems and connections to external production systems (such as EDOS and other ECS DAACs) via EBnet. At GSFC, EDC, and LaRC some Data Server hosts are contained on a dedicated FDDI ring in order to provide adequate bandwidth for DAAC-to-DAAC processing flow requirements. A dedicated FDDI ring provides access to the EBnet router to handle the DAAC-to-DAAC production flows. The FDDI Switch discussed in Section 4.10.2.1 is the central device connecting the FDDI rings together, and it provides the necessary routing and filtering control.

The User Network is an FDDI-based LAN connecting the users (via NSI, local campuses, general Internet, etc.) to the DAAC hosts responsible for providing user access. It has the main advantage of separating user and production flows. This allows DAAC processing data flows to be unaffected by user demand, so that even unanticipated user pulls do not hinder the Production Network. Users do not have access to any other hosts, such as Ingest or Data Processing devices. CSS and MSS servers are connected to the User Network but do not allow direct user access. These connections are required for communications with outside networks for such things as name lookups and receipt of Internet mail, as well as communication with and monitoring of the DAAC's interfaces to the user community (such as NSI and the local campus). The User

Network connects to NSI and the local DAAC campuses through an ECS router (discussed in Section 4.10.2.2) which provides the necessary routing and filtering controls.

The individual FDDI rings for both the User and Production Networks are implemented using FDDI concentrators to provide ease of wiring and central points of management. All DAAC hosts have FDDI interfaces and are attached directly to the FDDI rings. Workstations have single-attached FDDI cards, whereas the high-performance servers and processors on the Production Network have dual-attached FDDI cards to provide redundancy. The interfaces of these machines that are also on the User Network have single-attached interface cards. Dual-attached hosts are dual-homed to two separate FDDI concentrators. Printers, PCs, and x-terminals are connected to a FDDI ring via an FDDI-to-Ethernet hub.

The HIPPI Network interconnects Data Server hosts/devices and Science Processors in order to provide a high-speed network to handle the large data transfers between the two subsystems. The HIPPI network is implemented via a central HIPPI switch with switched interface ports, of at least 800 Mbps, connected directly to the high-powered processing and storage hosts. The HIPPI Network shifts the numerous transfers of large volumes of data onto a dedicated high-speed fabric.

4.10.1.2 SMC Network Architecture

The SMC network architecture, as illustrated in Figure 4.10.1.2-1, consists of two FDDI LANs connected to the GSFC DAAC ECS router. MSS and CSS servers are connected to one of the FDDI rings, and PC workstations and a printer are attached to an Ethernet network bridged to the FDDI ring via an Ethernet-to-FDDI hub. The Bulletin Board Server (BBS) and two FTP servers are attached to the second FDDI ring.

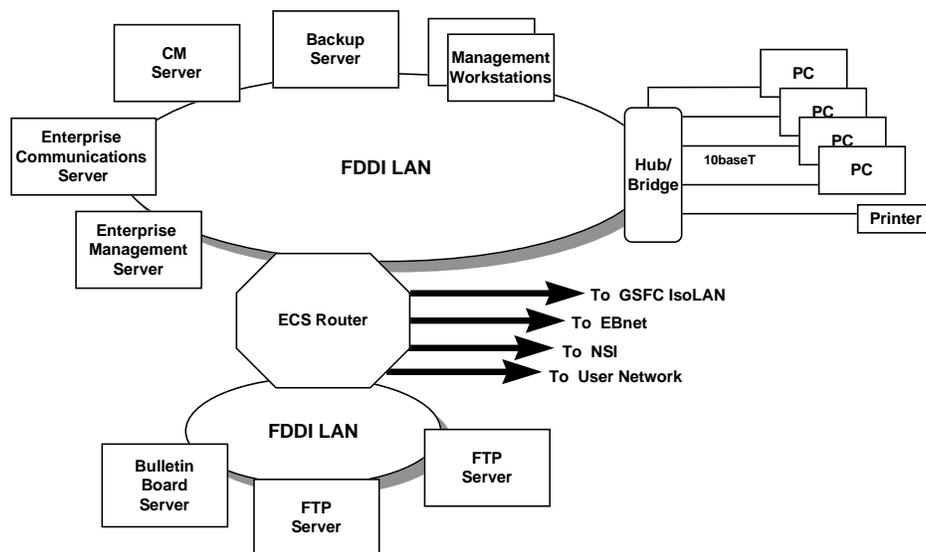


Figure 4.10.1.2-1. SMC Network Architecture Diagram

4.10.1.3 DAAC Addressing and Routing Architecture

The Planning and Data Processing, MSS, CSS, and Data Server/Data Management subsystems (collectively known as the Production Network) are connected to the FDDI switch on switched ports. They are assigned a Class C address space. The Ingest subsystem and the EBnet router are connected to the ECS FDDI switch on routed ports. They are assigned Class C or subnetted Class C address space. User Networks are connected to the ECS Router on routed ports. They are assigned Class C addresses. At GSFC, a subnet of the User Network Class C address space is used for the SMC. A subset of hosts (belonging to the Data Server and Processing subsystems) that are connected to the FDDI Production Network, also have interfaces connected to a HIPPI switch forming a HIPPI production network. The hosts are assigned private addresses as specified in RFC 1597 (as of 08/98). Documents that list IP address assignments to all hosts and network attached devices are listed in Table 4.10.1.5-1. All ECS address space (except for addresses used on HIPPI networks) is provided by EBnet from Class C address blocks designated by NSI.

Routing Information Protocol (RIP) is the protocol used to route IP packets within ECS as well as to/from external networks. ECS Production Networks are advertised to all ECS via EBnet. ECS ingest networks are advertised to data providers such as EDOS via EBnet. User Networks are advertised via RIP to NSI and campus networks.

4.10.1.4 Network-based Security Architecture

The network architecture provides basic levels of security to isolate and protect hosts and subsystems within the DAACs and SMC. Note that in addition to network-based security; ECS has implemented other security measures, such as DCE-based authentication and authorization, Kerberized telnet and FTP secure shell (SSH), and DCE access control lists (ACLs) which are discussed in CSS sections of this document.

At each ECS router connecting to external EOSDIS networks (such as EBnet) and external user networks (such as NSI), security filters have been implemented to control access to DAACs. These network and transport-layer filters control types of traffic that pass through the FDDI switch or ECS router, and they are able to control access to individual hosts as well as to whole subsystems.

4.10.1.5 Internetworking Subsystem Detailed Design

The ISS implementation level detailed design is documented in the documents listed in Table 4.10.1.5-1. All of the documents are under configuration control and can be obtained from ECS Configuration Management. The same information shown in Table 4.10.1.5-1 can be found at the WWW page <http://cmdm.east.hitc.com/baseline>. The documents are not on line for security reasons. Therefore special authorization is needed for their release.

Table 4.10.1.5-1. Internetworking Subsystem Baseline Documentation List

Document Name	EDC	GSFC	LaRC	NSIDC	SMC
Hardware/Network Diagram	921-TDE-002	921-TDG-002	921-TDL-002	921-TDN-002	921-TDS-002
Host IP Address Assignment Table	921-TDE-003	921-TDG-003	921-TDL-003	921-TDN-003	921-TDS-003
Network Hardware IP Address Assignment	921-TDE-004	921-TDG-004	921-TDL-004	921-TDN-004	921-TDS-004
Dual - Homed Host Static Routes	921-TDE-005	921-TDG-005	921-TDL-005	921-TDN-005	N/A
Ingest Host Static Routes	921-TDE-006	921-TDG-006	921-TDL-006	N/A	N/A

4.10.2 Network COTS Hardware

The DAAC and SMC LANs contain six types of COTS hardware: FDDI switches, Routers, Remote access servers/modems, FDDI concentrators, Ethernet hubs, and HIPPI switches. As described above, the FDDI rings within the DAACs are implemented via FDDI concentrators, and the FDDI switch is used to connect multiple Production Network FDDI rings together (refer to Figure 4.10.1.1-1). The FDDI-to-Ethernet hubs are used to connect PCs, printers, x-terminals, and remote access servers in the DAACs. At the SMC, the FDDI-to-Ethernet hubs are used to connect printers, x-terminals, and PC workstations. The Routers are used to provide access to external networks (NSI and Campus nets) via the User Network. The Remote access servers and modems provide access for instrument teams that want dial-up access at some of the DAACs. The HIPPI switches connect the Data Server and processing hosts with a high-speed fabric to be used for transferring large volumes of data between the two subsystems (Data Server and Data Processing). Table 4.10.2-1 provides a list of networking hardware used in ECS networks.

The following descriptions of Network Hardware devices are provided as illustrative detail. All details of the hardware configuration should be verified with the appropriate Hardware/Network Diagram shown in Table 4.10.1.5-1.

Table 4.10.2-1. Networking Hardware for ECS Networks (1 of 2)

Networking Hardware	Vendor
FDDI Switch	FORE PowerHUB 8000
Router (ECS Router)	Cisco 7507 (7513 at GSFC)
HIPPI Switch	Essential Communications EPS-16
Remote Access Server/Modems	Cisco 2509/Hayes OPTIMA 288, V.34.
FDDI Concentrator	Bay Networks 2914-04 concentrator with 12 M & 1 A/B port
Ethernet Hub	Cabletron MicroMMAC-22E; used for PCs, remote access sensors, printers and x-terminals
FDDI Cables	Multimode fiber cables with MIC connectors

Table 4.10.2-1. Networking Hardware for ECS Networks (2 of 2)

Networking Hardware	Vendor
Ethernet Cables	10baseT connection to printers, PCs, x-terms, printers and remote access servers
HIPPI Cables	Multi-wire copper cable for parallel HIPPI interface

4.10.2.1 ECS FDDI Switch

The ECS FDDI switch is the FORE PowerHUB 8000 with FDDI interface modules (DAS interfaces) and a powerful packet engine. The switch forms the core of the ECS Production network by interconnecting all FDDI segments that form the ECS production network as well as ingest segments. It also interfaces with EBnet. At the EOC, the FDDI switch interconnects the production, support and EOC M&O networks. All ports on the switch can be configured to switch or route giving the flexibility needed for configuring interfaces for data link layer or network layer connectivity.

The switch has redundant power supply and fan units. All interface modules are hot swappable.

4.10.2.2 ECS Router

The ECS Router is a Cisco 7500 series router (7513 at GSFC and 7507 at all other DAACs) running Cisco's Internetwork Operating System (IOS). All routers have Versatile Interface Processor (VIP) boards populated with FDDI DAS ports. The ECS Router is a key item of ECS DAAC networks in that it provides connectivity to the Internet via its interface with NSI. All ECS User Networks and M&O Networks at each DAAC are connected to the ECS Router. In addition, the SMC network is connected to the GSFC ECS Router.

The ECS Router has redundant power supply and fan units. All interface modules are hot swappable.

4.10.2.3 HIPPI Switch

The ECS HIPPI switch is an Essential Communications EPS-16 switch capable of supporting up to 16 parallel or serial HIPPI interface modules. It also has an Ethernet port for switch management. The HIPPI switch forms the core of the HIPPI fabric interconnecting Data Server and processing hosts providing capacity of up to 800 Mbps per connection. The ECS HIPPI switch is part of ECS DAAC networks at GSFC, EDC, and LaRC.

4.10.2.4 Remote Access Server and Modems

The ECS Remote Access Server (implemented at EDC) is a Cisco 2509 access server with 1 Ethernet and 8 asynchronous ports. It provides dial up access to instrument team members that need such service. Two Hayes OPTIMA 288, V.34 modems are attached to two asynchronous ports. The Ethernet port is used for connectivity to an Ethernet hub on the ECS User network.

4.10.2.5 Ethernet Hub

The ECS Ethernet hub (10BaseT) is a Cabletron MicroMAC-22E (with BRIM-F6 module) Ethernet-to-FDDI hub. It is a stackable hub with 1 A/B FDDI port and comes with 12 or 24 shared Ethernet ports. All ECS printers, x-terminals, PC workstations and remote access servers are connected to the Ethernet hub.

4.10.2.6 FDDI Concentrator

The ECS FDDI Concentrator is a Bay Networks System 2000 Model 2914-04. It is a stackable concentrator with 12 M Ports and 1 A/B Port (all MIC interfaces). All FDDI rings with multiple nodes on them are formed using several concentrators interconnected to form a ring. Ethernet hubs with FDDI uplinks are also connected to DAAC FDDI networks via the FDDI concentrators.

4.11 ECS General Process Failure Recovery Concepts

During ECS processing, client or server failures can occur. These failures cause certain recovery events to take place within the ECS. To understand the General Process Failure Recovery of the ECS processes, several key concepts must be described. These failure recovery concepts are:

- 1) DCE Rebinding
- 2) Sybase Reconnecting
- 3) Request Identification
- 4) Senior Clients
- 5) Request Responsibility
- 6) Queues
- 7) Request Responses
- 8) Duplicate Request Detection
- 9) Server Crash and Restart
- 10) Client Crash and Restart

These concepts compose the general philosophy of the ECS process failure recovery. The General Process is performed as a "process chain" to service requests for data or other services (e.g., order tracking or data retrieval from another processing system) in a client/server architecture. A brief description of each of the key concepts for General Process Failure Recovery follows.

4.11.1 DCE Rebinding

In the case of Distributed Computing Environment (DCE) client/server architecture, the clients call proxy objects that represent a server's server objects. A proxy object uses the dce name service to find its server object by its known name. The name service returns a dce internal reference to the server object, known as its "binding handle." The process itself is called "binding"

There are two possible failure situations to be discussed for rebinding. These failure situations are:

- System Startup Failures
- Server Crashes

It is conceivable that the initial attempt to "bind" with a server fails, for example, because the server is not up or because the dce name server is not running. The ECS infrastructure includes wrapper code for object-oriented distributed computing environment (oodce) that provides parameters for a number of automatic retries of a binding attempt and a retry interval.

The internal reference to a server object can become invalid, for example, if the server is shutdown and re-started. When this happens, the client needs to obtain a new reference. This process is called "rebinding." ECS client libraries contain code that makes an automatic attempt at rebinding with the server to support failure recovery.

Without such an automatic rebinding attempt, all client applications of a server would have to be brought down and re-started if a server fails (the re-started application can, of course, "bind" again). And if these applications have client applications, the client applications would need to be re-started and so on down the process chain. The result would be that the failure of a single server could ripple through most of the ECS and require the shutdown and re-start of a large portion of the system.

With automatic rebinding this is not usually the case. Only rarely will it be necessary to bring down a server to allow it to get a new "binding handle". When a server goes down (i.e., crashes), the other application(s), which communicate with it lose their "binding handle." However, the application(s) do continually try to rebind to the "downed" server. If the "downed" server comes back up before the number of retries are exhausted, the application(s) do eventually get a new valid "binding handle" for the re-started server and communications can continue. Operations may notice a brief pause in the execution of some applications, but as soon as the failed server is back on-line, the system reverts to a normal state.

DCE rebinding is generally done in the client library. Any configurable parameters are contained in the client libraries included in the client applications. The configurable parameters have defaults.

An example:

DDIST calls the STMGT Staging Disk Server to allocate disk space. If the Staging Disk Server crashes and has to be restarted, the binding handle DDIST has for it is invalid. DDIST cannot use the binding handle in the future (even after the server comes back up). However, the Staging Disk Server client library automatically discovers it lost the connection to its server and tries to rebind. The rebinding succeeds when the server comes back up and is ready to accept requests again. Distribution requests requiring services from this particular Staging Disk Server would have paused briefly, but would have been resumed automatically, without a need to restart DDIST.

4.11.2 Sybase Reconnecting

A similar approach has been implemented by the ECS infrastructure that provides the interface with the Sybase Servers. For example, an ECS application may attempt to connect to its Sybase server while that server is still in the process of starting up. The connection attempt fails, but the infrastructure code attempts the connection for a configurable number of times, waiting for a configurable amount of time between each connection attempt.

Most ECS applications obtain a connection for the duration of a transaction and relinquish it when they are done with it. These applications have been directed to implement the following recovery behavior: if they get a Sybase error that requires the transaction be re-done, they release and then re-request the connection. If the cause of the error was a Sybase server fault, this

connection attempt fails, but causes the infrastructure code to enter the connection re-try loop. If the Sybase server is restarted before the retries are exhausted, the application continues normally and now completes the transaction that was in progress when the Sybase fault occurred.

However, operations should be aware of the following facts:

- Not all ECS applications are able to use the ECS Sybase interface code. For example, the Science Data Server does not, for performance reasons.
- Not all ECS applications are able to use Sybase transactions and automatic re-connection in the manner described. For example, the PDPS software needed to follow a different approach to work around Sybase internal deadlocking.

4.11.3 Request Identification

ECS generates a unique identifier for each type of request that requires such fault handling provisions. These "recoverable requests" fall into one of these two categories:

- 1) User Requests: their request identifiers are generated by the System Management Subsystem (MSS) when request tracking information is created
- 2) System Requests: their identifiers are system generated and referred to as RPC ID. (RPC refers to Remote Procedure Call, the DCE mechanism by which requests are submitted from client to server.). They are based on the Universal Unique Identifier (UUID), a DCS mechanism for creating unique identifications.

Some examples of ECS processes that use the User Requests are the V0 Gateway, E-mail Parser Gateway, and ASTER Gateway. Some examples of ECS Computer Software Configuration Items (CSCIs) that use the System Requests are PLANG and PRONG.

The following describes how request identification is used during recovery. As a request propagates through the system, each associated client/server exchange is assigned a unique RPC ID. However, the RPC ID for each interaction is derived from the previous RPC ID received by the client for this request. Thus, all RPC IDs associated with a given request have a common portion that relates the various client/server calls to one another. More importantly, given the previous RPC ID, clients consistently reproduce the same RPC ID that was submitted to the server on the subsequent event. The concept of reproducible RPC IDs is central to the ECS fault recovery capability. When requests are retried from client to server, they are always submitted with the same RPC ID as was used in the original submission of the request, even if either the client or server has crashed between retries.

RPC IDs are also central to the check-pointing aspect of fault recovery. As requests arrive at fault recovery-enabled servers, they are recorded in a persistent store (typically, a database), tagged with the RPC ID which identifies the request. As the request is serviced, check-pointing state information may be updated in the persistent store, up to and including the completion status of the request. This allows the servers to resume servicing from the last check-pointed state, particularly upon re-submission from a client.

Many kinds of requests do not pose recovery issues and thus, do not employ request identifiers. For example, if a search is submitted to the Science Data Server, and no response is received, the client

application can simply re-submit the search. However, some types of requests do pose recovery issues. For example, if PDPS inserts the output of a PGE execution into the archive, there needs to be some guarantee that the output granules are not lost when faults occur.

4.11.4 Senior Clients

A Senior Client is an ECS client process that originates an ECS request that has fault recovery requirements and may lead to a chain of sub-requests. The Senior Client assigns the original request identifier (rpcid). It is responsible for re-submitting the request if it gets a retry error or no response. It is responsible for reassigning the same rpcid upon re-submission of a request.

Senior Clients include the Ingest Granule Server, the PDPS Queuing Server, the ASTER Gateway, the V0 Gateway, the E-mail Parser Gateway, the On-demand Processing Request Manager (ODPRM), the Subscription Server, and the Science Data Server.

Senior Clients that send requests and receive acknowledgments of receipt of their requests from the receiving servers can expect to receive an outcome (a response, a code, data, or messages). If no acknowledgments are received from the receiving server, the Senior Client must re-submit the request with the same RPCID as the initial request after a failure recovery. The unique RPCID helps receiving servers to recognize duplicate requests so these duplicate requests can be acknowledged or ignored.

There is one exception to this re-submission rule. Senior Clients are not responsible for recovery of the process environment and completion of the requests, if they are cold started. If the restart is a cold start, there are no automatic restarts for any previous requests and all requests are submitted as new requests.

In essence, a Senior Client takes on the role of the "end user" for system requests. If anything happens to a request upstream, it has the ultimate responsibility for deciding what to do with the request (retry or suspend/abort it and tell the operator, i.e., "the buck stops" with the Senior Client).

4.11.5 Request Responsibility

The responsibility for the handling recoverable requests by a server is given in the ECS by determining if the request is synchronous or asynchronous.

- **Synchronous Requests**

On a synchronous request, the application that submits the request will be waiting for a response. Regardless of how the request is handled downstream, whether it succeeded or failed depends on the response that the waiting application gets back. From its perspective, the request is not complete until it receives a response.

Therefore, if an ECS application initializes a request and submits it synchronously, it has the responsibility for getting the request completed. This means that if the request does not complete, for example, because the connection is lost to the server, which the request is submitted to, the application needs to submit it again.

ECS examples of synchronous interfaces include Order Tracking, User Profile updates, Data Dictionary and Advertising updates, and Data Processing Subsystem staging and

destaging (acquire and insert) requests to the Science Data Server. Applying the above rules, this means the Data Processing Subsystem has the ultimate responsibility for ensuring that destaging (archive inserts) completes successfully. If for some reason, a request does not complete, it retries the archive insert. If after a sufficient number of retries (the number is configurable) the insert was not successful, the processing job is placed on hold so the operator can investigate. Also, staging and destaging requests involve many ECS servers. Their recovery must be coordinated across these servers. Therefore, the Data Processing Subsystem acts as a "Senior Client" and assigns a unique identification to each request.

- **Asynchronous Requests**

When an application sends an asynchronous request, the receiving server is responsible for completing the request once it accepts the request. For example, the server may need to save the request (perhaps in a queue in a database) before sending an acknowledgment to the originating application. Of course, the server (Server A) can eventually complete processing the request and pass it on to another server (Server B), also asynchronously. Once Server B accepts the request, it is responsible for seeing it to completion.

ECS examples of asynchronous interfaces include the V0 Gateway order (ACQUIRE) interface with the Science Data Server, and the corresponding acquire interfaces of the E-mail Parser Gateway, and ASTER Gateway. The SDSRV, on the other hand, interfaces with DDIST synchronously for its synchronous requests (e.g., from PLANG and PRONG), and asynchronously for its asynchronous acquire requests.

For example, when a user submits an order via the EDG client to the V0 Gateway, the gateway turns that into an asynchronous acquire request to the Science Data Server (SDSRV). As soon as the SDSRV accepts the request, the V0 gateway returns control to the EDG and the user. Let's assume this is a Landsat subsetting request. The SDSRV is now responsible for the request and completing the subsetting (if at all possible). It does so by saving the request in the SDSRV database. However, once subsetting is complete, the SDSRV submits the result for distribution to Data Distribution (DDIST), also asynchronously. Once DDIST accepts it, it has the responsibility for it (DDIST saves it in the distribution queue in the DDIST database).

4.11.6 Queues

The reason queues are mentioned here is because they represent an important aspect of recovery. If a server uses queues to defer work until later, it needs to be concerned about what happens to the queue if the server crashes. The recovery rules in the request responsibility section state:

- if the server queues up synchronous requests, the client application is responsible for recovering the synchronous requests
- if the server queues up asynchronous requests after accepting them, it is responsible for the asynchronous requests, which means, if a queue contains asynchronous requests, the server must make sure that it can recover the queue in case of a crash

DCE queues require special consideration. If there are more requests than the server has threads to process, DCE queues up the requests. However, DCE does not recover this queue in case of a restart after a fault. The server does not know about these requests: it is not aware of these requests until DCE takes them out of the queue and hands them over to the server.

If a server crashes with a number of requests stuck in the DCE queue, their clients do not receive a response to the request submissions. The clients only receive a dce error telling them that the connection to the server was lost. As a result, the client application does not know for certain what happened to its request.

To summarize, DCE queues are not recoverable. When an ECS server crashes, any rpcs that were queued by DCE are lost without ever having been seen by the server, and the client does not receive a server response (only a DCE error). Therefore, DCE queues are not a viable mechanism for keeping requests queued: the client is left in limbo as to whether the request was actually received; and the queue cannot be restored when the server is restarted.

Instead, a server handling asynchronous requests must keep a queue in a safe place so it can recover in case of a restart (such a restart that recovers the current requests is called a "warm start"). If a warm start takes place with asynchronous requests, the sending application does not even notice that there was a problem. The processing gets completed eventually.

Note, however, that queued synchronous requests require special consideration: If a warm start takes place and some of the queued requests are synchronous, the sending application is generally aware of the failure (it had to rebind, See DCE Rebinding Section). Since it did not receive a response, it re-submits the request. The server must recognize the request as a re-submission and either ignore it or - if it already completed by the time the re-submission is received - return the completion status as a response to this rpc. Moreover, servers that might handle a large number of concurrent synchronous requests have to be able to deal with a sudden spike of request submissions following a warm start, as their clients re-submit these requests.

A warm start can cause a problem; for instance, one of the active requests may be the reason the server crashed. This could result in a warm restart loop: each time the warm start is attempted the server crashes again because of the bad request. In such a case, operations can use a cold start to empty the queue of all requests (at the expense of having to recover queued asynchronous requests that were lost manually).

4.11.7 Request Responses

Servers have the responsibility to classify a response appropriately. Client applications have the responsibility to process a response appropriately, depending on its type.

Client applications can pass the response on to the calling application (e.g., success, warning, or fatal error); or retry (retry error). At the beginning of a request chain, there may be a user or operator (if this is a user or operator submitted request). In that case, the error is passed back to the user/operator for action where possible.

Where this is not possible (e.g., system generated requests, or if a data order runs into an error after it was already accepted and the user/operator is no longer connected), errors are brought to the attention of the DAAC operations staff for action.

Failure events are classified as having any of three severity levels:

- fatal errors,
- retry errors and
- warnings

Fatal errors are returned when a request cannot be serviced, even with operator intervention. For example, if a request is made to distribute data via FTP to a non-existent host, the request will be failed with a fatal error.

Retry errors can be recovered from, and such errors should be returned back to the client only when the server cannot recover from the error automatically. Retry errors may also necessitate operator assistance for recovery purposes, such as in the case of a tape that is left in a device and must be manually removed.

Warnings are provided where operations can proceed without interruption, but where an unexpected circumstance was detected. For example, if a client requests that a file is removed, and the file does not exist, there is no error, per se, but a warning is generated to caution the client that the file to be removed did not exist in the first place.

The situation where a server does not return a response represents a special case. It can occur, for example, when an application calls a server and the server crashes before it can send a response or there is a communication error that prevents a response within a reasonable time. The situation is important because now the client application does not really know what happened to the request:

- a. did it reach the server?
- b. did the server start the request but not complete it?
- c. did the server complete the request with an error but was not able to send the error response?
- d. did the server process it successfully?

The ECS recovery policy is that in such situations, the client application should either

- a. re-submit the request unless
- b. it is possible to return an appropriate error to the user/operator who submitted the request (to avoid leaving them with a hanging GUI while ECS goes through endless retries)

Note that if the request did reach the server, the server now sees the request twice (i.e., this has become a duplicate request). Therefore, there need to be provisions to handle duplicate requests gracefully.

Table 4.11.7-1 summarizes the five categories of request responses, and the specific requirements for the application or server that is currently responsible for the request. ECS servers have been directed to classify their responses accordingly.

Table 4.11.7-1. Request Responses

Request Response	Response Description
Success	The server sends back a message to acknowledge the successful completion of the request to the client. The request is considered complete.
Warning	This is provided where operations proceed without interruption, but where an unexpected circumstance is detected. The calling application needs to determine whether to alert the user or operator of the situation.
Error, retry the request	This can happen if the server encountered a temporary error condition, such as a media error on output. The request can be "retried" and the application responsible for the request should re-submit it after a suitable wait time. However, if the request does not succeed after a (configurable) number of retries, it should be considered "failed." If the application is supported by a GUI, the request may be suspended (if it makes sense to alert the operations staff to remedy the situation).
Error, cannot retry the request	This can happen if the server encounters an error condition that is sure to re-occur if the same request is submitted again. Examples might be a syntax error in the request (indicating some internal software problem), or an attempt to retrieve a non-existent granule. The request is considered "failed." The server responsible for the request sends back a failure notification. If the application is supported by a GUI, the request may be suspended (if it makes sense to get the operations staff involved at this point), but the operations staff may or may not be able to help.
No response returned by server	This can happen, for example, if the server to which the request was submitted crashes before a response or an acknowledgment is returned. In this case, the client can make no assumptions about the request. The client responsible for the request should send the request again or retry the request.

Transient errors such as network errors are always retry errors. In general, clients and servers that experience transient, retry errors can first attempt to recover by retrying the operation automatically. One special case of this is "rebinding." Rebinding refers to the process by which a client automatically attempts to re-establish communications with a DCE server in the event that communications are disrupted. This disruption may be caused by transient network failure, or by the server being brought down or crashing. In any case, the client will automatically attempt to reconnect to the server for a period of time that is configurable on a client-by-client basis.

ECS processes that encounter an error or receive an error from a server request may either pass the error back to a higher-level client or present it to the operator for operator intervention. The fault handling policies are detailed in Table 4.11.7-2:

Table 4.11.7-2. Fault Handling Policies (1 of 3)

CI	Client Process(es)	Fault Handling Policy
PLANG, PRONG	EcPISubMgr	<p><u>Retry errors:</u> All Subscription processing errors are retried a configurable number of times and for a configurable time period. After these number of times (or time period) the subscription notice is lost.</p> <p><u>Fatal errors:</u> the error is logged, but the subscription notice is lost. The operator can provide it manually later, after the cause of the failure has been corrected.</p>
	EcPIPREGui EcPIWb	<p>Retry errors: Since these are GUI applications, errors are reported to the user and it is his/her responsibility to retry the request.</p> <p>Fatal errors: Errors are reported to the user.</p>
	EcDpAtStageDAP EcDpAtInsertTestFile EcDpAtInsertStaticFile EcDpAtInsertExeTarFile EcDpAtSSAPGui EcDpAtGetMCF	<p>Retry errors: Some automatic retries of requests exist, but in general these are command line tools and as such report any errors to the user and it is his/her responsibility to retry the request.</p> <p>Fatal errors: The User is sent a fatal error message.</p>
	EcDpPrDM EcDpPrEM	<p>Retry errors: Errors are retried a configurable number of times, then the job is failed and it is up to the Production Monitor to restart the job through AutoSys.</p> <p>Fatal errors: A fatal error message is logged. It is up to the Production Monitor to correct the problem and then restart the job through AutoSys</p>
	EcDpPrJobMgmt	<p>Retry errors: Requests are NOT retried to EcDpPrDeletion. This can result in the loss of Interim Granule delete notifications (interim granules may not be deleted from the system).</p> <p>Fatal errors: N/A</p>
	EcDpPrDeletion	<p>Retry errors: No retries are implemented. Status from DSS is <u>not</u> checked. This can result in missed granule deletions.</p> <p>Fatal errors: N/A</p>
	EcPIOdMgr	<p>Retry errors: Retries errors from the Science Data Server and the Subscription Server.</p> <p>Fatal errors: Logs errors and terminates current on demand requests.</p>

Table 4.11.7-2. Fault Handling Policies (2 of 3)

CI	Client Process(es)	Fault Handling Policy
INGST	EcInGran	<p>Retry errors: Errors are retried a configurable number of times, then the granule is suspended. The operators can then either cancel or resume the suspended granule from the Ingest GUI.</p> <p>Fatal errors: The granule is failed. Granule failures are displayed on the Ingest GUI. The external system originating the Ingest request may be notified (depending on the ingest protocol).</p>
	EcInReqMgr	<p>Retry errors: Errors connecting to EcInGran are retried forever. Retry errors involving staging disks are retried a configurable number of times, then the request is failed.</p> <p>Fatal errors: Errors are failed immediately. The external system originating the Ingest request may be notified (depending on the ingest protocol).</p>
	EcInGUI	<p>Retry errors: Any error results in the request failing.</p> <p>Fatal errors: Any error results in the request failing.</p>
	EcInPolling EcInAuto	<p>Retry errors: Errors are retried forever, with a delay between retries.</p> <p>Fatal errors: Errors are failed immediately, and are displayed on the Ingest GUI.</p>
	EcInEmailGWServer	<p>Retry errors: N/A</p> <p>Fatal errors: E-mail that cannot be processed is moved to a failed directory, but no operator notification is provided.</p>
SBSRV	EcSbSubServer	<p>Retry errors: Errors are retried for a configured amount of times, then suspended. The operators can then either cancel or resume the suspended acquire requests through system provided scripts.</p> <p>Fatal errors: are treated like retry errors.</p>

Table 4.11.7-2. Fault Handling Policies (3 of 3)

CI	Client Process(es)	Fault Handling Policy
SDSRV	EcDsScienceDataServer EcDsHdfEosServer	<p>Retry errors: Errors are retried a configurable number of times, then passed back to the calling client process unchanged. The default retry policy for SDSRV servers is “retry forever.” For async Acquire requests that involve subsetting, retry errors encountered with the HDF servers are not returned to the client. Instead the request is queued for future execution.</p> <p>Fatal errors: Errors are passed back to the calling client process if the request was synchronous. Errors associated with asynchronous requests are logged but do not appear on any GUI. The request itself is lost.</p> <p>After a SDSRV fault, HDF servers must be restarted manually by the Operator.</p>
DDIST	EcDsDistributionServer	<p>Errors are presented to the operator via the DDIST GUI.</p> <p>Retry errors: Errors are presented as “Suspended with Errors” and can be resumed by the operator.</p> <p>Fatal errors: Errors are presented as “Failed.” For synchronous requests, fatal errors are also passed back to the calling client process. For asynchronous requests, fatal errors are returned to the requesting user via e-mail notification.</p>
STMGT	EcDsStArchiveServer EcDsStStagingDiskServer EcDsStStagingMonitorServer EcDsStPullMonitorServer EcDsStFtpDisServer EcDsStIngestFtpServer EcDsSt8MMServer EcDsStD3Server EcDsStCDROMServer EcDsStPrintServer	<p>Retry errors: Errors are passed back to the calling client process.</p> <p>Fatal errors: Errors are passed back to the calling client process.</p>

4.11.8 Duplicate Request Detection

The above scheme for handling requests in cases of faults poses a potential problem. The request could have been re-submitted because there was no response returned by the server. But, in fact, the server completed the request but was unable to get the status back to the client (e.g., because of communications problems or a machine crash). The following measures are intended to deal with this situation:

- **Trivial duplicate requests.** There are many interfaces where sending a new request to retry a service whose outcome is unknown either has no or negligible impact on the ECS. This is because many ECS services have been designed with this goal in mind. For example, after a failure, the Science Data Server CSCI can send a duplicate request for inserting a new collection to the Data Dictionary CSCI or the Advertising Service CSCI. The Data Dictionary CSCI or the Advertising Service CSCI simply interprets the second request as an update for the (now) existing collection. When the SDSRV exports the same event more than once to the Subscription Service Computer Software Component (CSC), it assumes it is meant as a replacement for the previous one. This made designing the recovery for an ESDT update fairly simple. If the update fails, it can always be re-started at the beginning. Any duplicate requests issued to dictionary, advertising, or subscription services are of no consequence.
- **Recognize non-trivial duplicate requests.** Where executing the same request more than once can have undesirable consequences, ECS provides a mechanism for recognizing re-submitted requests. Each request is tagged with a unique identifier (see Request Identification Section). Upon submission of a request, the receiving server of the request must check the identifier and recognize when it is a re-submission of a previous request it received. For example, the server may realize that the request has been completed and simply acknowledges the successful completion. This is what happens, for example, when the Data Distribution CSCI recognizes when it receives a duplicate staging request from the PRONG CSCI (data processing) and sends back an acknowledgment of receipt of the duplicate request but otherwise, ignores it. Another example is the STMGT CSCI recognizing a duplicate insert request (perhaps because some ingest server was restarted) and ignoring it because it already completed. This last case is an example where the sub-identifier is important. The insert requests are actually issued by the SDSRV. But SDSRV, rather than trying to recognize the duplicate insert request, simply creates the corresponding archive requests and passes them on to Storage Management with the same sub-identifiers as before. The SDSRV relies on Storage Management to sort out how far the overall insert operations progressed.

4.11.9 Server Crash and Restart

- **Server Crash**

When a server crashes, the only impact on the system is that clients cannot continue to submit requests for processing. Synchronous requests that are in progress will result in a DCE exception being thrown back to the client process, which will enter a rebinding failure recovery mode (see DCE Rebinding section above). Attempts to submit requests while the server is down will result in the client blocking until a communications timeout has been reached.

- **Server Restart**

When a server restarts, it may perform various re-synchronization activities in order to recover from an unexpected termination. In the event of a server cold start or cold restart, the server will also cancel all outstanding requests and reclaim all associated resources.

Note that the distinction between cold start and cold restart is described in the section above on Start Temperature. Specifics of server startup behavior are detailed in Table 4.11.9-1. Unless otherwise stated, existing request queues are always retained for warm restarts and cleared for cold starts or cold restarts.

Table 4.11.9-1. Server Response versus Restart Temperature (1 of 3)

CI	Server(s)	Special Behavior on Warm Restart	Special Behavior on Cold Start or Cold Restart
PLANG, PRONG	EcDpPrJobMgmt	Jobs in AutoSys and jobs waiting in the queue are read from the database. Any jobs that are ready are placed into AutoSys from the queue if there are processing slots available.	N/A
	EcPISubMgr	Any subscriptions that have not been processed are read from checkpoint file and processed.	N/A
	EcDpPrDeletion	Interim granules marked for deletion are read from the database and will be deleted when time out occurs.	N/A
INGST	EcInGran	The EcInGran server automatically restarts submitted requests from the beginning. If a file has been FTP'ed, it will not re-do the FTP of that file.	All granule requests are cancelled. Existing request queues are cleared for cold start and retained for cold restart.
	EcInReqMgr	EcInReqMgr re-synchs requests that are in progress with EcInGran, and resumes processing from the last check-pointed state.	On cold start, all active requests are moved to the summary tables. On cold restart, each granule is re-submitted to the EcInGran where it is failed. EcInReqMgr then re-submits the request to EcInGran, where it is processed as a new request. Existing request queues are cleared for cold start and retained for cold restart.

Table 4.11.9-1. Server Response versus Restart Temperature (2 of 3)

CI	Server(s)	Special Behavior on Warm Restart	Special Behavior on Cold Start or Cold Restart
	EclnPolling	Re-submit requests that were in progress at the time of fault. Continue polling for remaining requests in polling directory.	Cleans up files and terminates any requests which had not yet been sent to EclnReqMgr. Requests remaining in the polling directory are sent as new requests.
	EclnAuto	Re-submit requests that were in progress at the time of fault. The external client may re-submit the request - this can lead to duplicate ingest.	Cleans up files and terminates any requests which had not yet been sent to EclnReqMgr.
	EclnGUI EclnEmailGWServer	N/A	N/A
SBSRV	EcSbSubServer	SBSRV will perform all unprocessed actions (including re-submissions of ACQUIRE requests to the SDSRV), and resume accepting new event notifications from the SDSRV.	SBSRV will remove all unprocessed requests.
SDSRV	EcDsScienceDataServer EcDsHdfEosServer	Restart Async Acquire Requests that were in progress before the crash. (Note that the queue of asynchronous acquire requests is retained. Assumption: Synchronous requests are re-submitted by the respective senior client applications (PRONG, INGST). Send event notifications to SBSRV for any services completed before the crash for which a subscribed event is registered and has not been sent to SBSRV.	Purge the queue of Async Acquire Requests. Purge the queue of SBSRV Event Notifications.

Table 4.11.9-1. Server Response versus Restart Temperature (3 of 3)

CI	Server(s)	Special Behavior on Warm Restart	Special Behavior on Cold Start or Cold Restart
DDIST	EcDsDistributionServer	Request Processing is restarted from the last check-pointed state.	On cold start, STMGT subsystem is informed of a cold start, and the EcDsDistributionServer will delete all (prior) request information from its databases.
STMGT	EcDsStArchiveServer	Retains existing request queues.	For partially completed Store requests, the files copied into the archive are removed. For partially completed Retrieve requests, the access count is decremented in the Read-Only Cache.
	EcDsStStagingMonitorServer	The contents of the Read-Only Cache are synchronized with the database. Discrepancies are logged and removed.	All files are removed from the Read-Only Cache. Links to files in the Read-Only Cache are left dangling.
	EcDsStStagingDiskServer	The set of staging disks in the staging area is synchronized with the database. Discrepancies are logged and removed. Existing request queues are cleared.	All staging disks are removed.
	EcDsStPullMonitorServer	The contents of the Pull Area and user request areas are synchronized with the database. Discrepancies are logged and removed.	All files in the Pull Area and all user request areas are removed.
	EcDsStFtpDisServer EcDsStIngestFtpServer	Existing request queues are retained.	Existing request queues are cleared.
	EcDsSt8MMServer EcDsStD3Server EcDsStCDROMServer EcDsStPrintServer	N/A	N/A

- **Request Re-submission**

Upon restarting a crashed client or server, requests are typically re-submitted. If the restarted process was started warm, the fault recovery capabilities permit the server to resume processing of the request from its last check-pointed state. This prevents needless repetition of potentially time-consuming activities. Specific behavior of servers upon re-submission of a request is detailed in Table 4.11.9-1. Note that a cell value of N/A means that the server either has no clients or that the clients do not re-submit requests.

Table 4.11.9-2. Server Response for Request Re-submission (1 of 2)

CI	Server(s)	Behavior on Request Re-submission
PLANG, PRONG	EcDpPrJobMgmt	Requests are submitted synchronously. If the entire request is re-submitted by a client then only that part of the re-submitted request that hasn't been completed will be re-processed.
	EcDpPrDeletion	Requests are submitted synchronously. If the entire request is re-submitted by a client then only that part of the re-submitted request that hasn't been completed will be re-processed.
INGST	EcInGran EcInReqMgr EcInPolling EcInAuto EcInGUI EcInEmailGWServer	N/A
SBSRV	EcSbSubServer	When SDSRV re-submits the same request, if SBSRV received and buffered it successfully, this second request will not be processed. Instead, SBSRV just returns a successful status to the client. When SBSRV re-submits the same request to its action provider, SDSRV, it will use the same rpc ID for this request. As long as SDSRV returns a successful status, this request will be removed from SBSRV side and will not be re-submitted.
SDSRV	EcDsScienceDataServer EcDsHdfEosServer	All requests are serviced as if they are new requests. Note that since RPC IDs are generated automatically and reproducibly, SDSRV will typically recreate the same allocation requests on a re-submission. This can trigger special logic to handle requests for which an allocated staging disk has been transferred to DDIST. See the cell below for request re-submission behavior for EcDsStStagingDiskServer.

Table 4.11.9-2. Server Response for Request Re-submission (2 of 2)

CI	Server(s)	Behavior on Request Re-submission
DDIST	EcDsDistributionServer	If previously submitted and completed, the request status is returned based on the check-pointed request status. Otherwise, the client request thread is synchronized with the worker thread that is actually servicing the request.
STMGT	EcDsStArchiveServer	The request is restored from the last check-pointed state. For Store requests, copies into the archive are resumed from the last file copied. For Retrieve requests, the entire Retrieve request is reprocessed. However, files previously retrieved for the request are, in all likelihood, still in the read-only cache.
	EcDsStStagingMonitorServer EcDsStFtpDisServer EcDsStIngestFtpServer	If previously submitted and completed, the request status is returned based on the check-pointed request status. Otherwise, the request is processed anew.
	EcDsStStagingDiskServer	For staging disk allocation, a warning is returned to the client if the client re-submits the allocation request under which the disk was created, but the staging disk has been transferred to another process and/or destroyed. Specifically, SDSRV creates staging disks, which are subsequently transferred to DDIST (via the Claim Ownership interface) and released by DDIST. If SDSRV attempts to recreate the same staging disk by re-submitted the allocation request with the same RPC ID, no staging disk is returned, and SDSRV is sent the warning that the staging disk has already changed ownership. SDSRV then skips ahead to re-submit its request to DDIST.
	EcDsStPullMonitorServer EcDsSt8MMServer EcDsStD3Server EcDsStCDROMServer EcDsStPrintServer	The re-submitted request is processed as if it were a new request.

4.11.10 Client Crash and Restart

- **Client Crash**

When a client crashes in the ECS system, fault recovery-enabled servers have several possible responses. Servers may continue to service client requests, independent of the client's status. Servers may choose to suspend processing of client requests, but permit the requests to be resumed upon client recovery. Or, servers may terminate servicing of the client requests, canceling all work done on the requests. The behavior of each CI is detailed in Table 4.11.10-1. Note that the behavior of a server in the event of a client crash does not vary from client to client.

Table 4.11.10.1. Server Responses to Client Failures

CI	Server(s)	Behavior on Client Crash
PLANG, PRONG	EcDpPrJobMgmt EcPISubMgr EcDpPrDeletion	Requests in process are serviced to completion
INGST	EcInGran EcInReqMgr EcInAuto	Requests in process are serviced to completion.
	EcInGUI EcInPolling EcInEmailGWServer	N/A
SBSRV	EcSbSubServer	Since its client, SDSRV, is also the action provider of SBSRV, SBSRV will proceed to finish all triggered subscriptions till the point that SDSRV has to be called. By then, all requests are stored for later retry.
SDSRV	EcDsScienceDataServer EcDsHdfEosServer	Requests in process are serviced to completion.
DDIST	EcDsDistributionServer	Requests in process are serviced to completion.
STMGT	EcDsStArchiveServer EcDsStStagingMonitorServer EcDsStPullMonitorServer EcDsStFtpDisServer EcDsStIngestFtpServer EcDsSt8MMServer EcDsStD3Server EcDsStCDROMServer EcDsStPrintServer	Requests in process are cancelled by thread rundown.
	EcDsStStagingDiskServer	Requests in process are cancelled by thread rundown. Non-persistent staging disks allocated by the client are leaked (NCR 15262 – will be fixed in 6A).

- **Client Restart**

When a client restarts in the ECS system, it sends a restart notification to each server with which it interacts. Clients notify servers that they have come up “cold” or “warm”, and do not differentiate between cold start and cold restart. Generally, the notification temperature sent to the server matches the temperature at which the client process is restarted.

Table 4.11.10-2 shows exceptions to the general behavior for client submission of restart notification:

Table 4.11.10-2. Client Restart Notification Exceptions (1 of 2)

Client Process(es)	Server Process(es)	Restart Notification	
PDPS		N/A	
EclnGran	EcDsScienceDataServer EcDsStStagingDiskServer	Matches start temperature (Also see Note 1 below)	
	EcDsStIngestFtpServer	N/A (not supported by server)	
EclnReqMgr	EcDsStStagingDiskServer	Matches start temperature (See Note 1 below)	
EclnGUI	EcDsStStagingDiskServer	Always sent warm (See Note 1 below)	
	EcDsStD3Server	N/A (not supported by server)	
EclnPolling EclnAuto EclnEmailGWServer	N/A	N/A	
EcSbSubServer	EcDsScienceDataServer	Match start temperature	
EcDsScienceDataServer	EcDsDistributionServer	Always sent warm	
	EcDsStArchiveServer	Always sent warm (Also see Note 1 below)	
	EcDsStStagingDiskServer	Always sent cold (Also see Note 1 below)	
EcDsHdfEosServer	EcDsStStagingDiskServer	Sent cold by default (Also see Note 1 below)	
EcDsDistributionServer	EcDsStArchiveServer EcDsStStagingDiskServer	Matches start temperature (Also see Note 1 below)	
	EcDsStFtpDisServer EcDsSt8MMServer EcDsStD3Server EcDsStCDROMServer EcDsStPrintServer	N/A (not supported by server)	
	EcDsStArchiveServer EcDsStStagingMonitorServer EcDsStStagingDiskServer	Matches start temperature (Also see Note 1 below)	
	EcDsStStagingMonitorServer EcDsStPullMonitorServer	N/A	
	EcDsStStagingDiskServer	EcDsStStagingMonitorServer EcDsStStagingDiskServer	Matches start temperature (Also see Note 1 below)
	EcDsStFtpDisServer	EcDsStStagingDiskServer EcDsStPullMonitorServer	Matches start temperature (Also see Note 1 below) N/A (not supported by server)

Table 4.11.10-2. Client Restart Notification Exceptions (2 of 2)

Client Process(es)	Server Process(es)	Restart Notification
EcDsStIngestFtpServer EcDsSt8MMServer EcDsStD3Server EcDsStCDROMServer EcDsStPrintServer	EcDsStStagingDiskServer	Sent cold by default (See Note 1 below)

Note 1: Restart notifications sent to the EcDsStArchiveServer and EcDsStStagingDiskServer servers are sent as needed. Since multiple server instances exist for each, a restart notification is sent on the first attempt to contact each instance. Note that the restarted client may not connect to a particular server instance (e.g., the WKS Archive Server) for an extended time after restarting. A restart notification will still be sent on the first connection, regardless of how long the client process has been running. Also note that restart notifications to the EcDsStArchiveServer and EcDsStStagingDiskServer servers are sent automatically, even if the client has not explicitly issued a restart notification. Automatically generated notifications are sent as cold restart notifications by default, if the client issues no explicit restart notification.

The default server behavior in response to a startup notification from a client is as follows:

- **Warm Notification:** Outstanding requests for the restarted client are left available in the persistent store. These requests may be re-submitted by the client, and will be serviced to completion upon re-submission. Associated resources are left allocated until the requests are completed.
- **Cold Notification:** All outstanding requests for the restarted client are cancelled. If the client re-submits any cancelled request using the same RPC ID (e.g., by pressing the Retry button from an operator GUI), it will be failed with a fatal error due to the client cold startup notification. Any resources associated with the cancelled requests are released and reclaimed by the system.

Specific aspects of server behavior upon receipt of a client restart notification are detailed in Table 4.11.10-3:

Table 4.11.10-3. Server Responses to Client Notification

CI	Server(s)	Behavior on Cold Notification	Behavior on Warm Notification
PLANG, PRONG	EcDpPrJobMgmt EcPISubMgr EcDpPrDeletion	N/A	N/A
INGST	EclnGran EclnReqMgr EclnPolling EclnAuto EclnGUI EclnEmailGWServer	N/A	N/A
SBSRV	EcSbSubServer	N/A	N/A
SDSRV	EcDsScienceDataServer	N/A	N/A
	EcDsHdfEosServer	N/A	N/A
DDIST	EcDsDistributionServer	General	General
STMGT	EcDsStArchiveServer	For partially completed Ingest operations, all files stored are removed. (Partial granules are never permitted in the archive.)	General
	EcDsStStagingMonitorServer EcDsStPullMonitorServer EcDsStFtpDisServer EcDsStIngestFtpServer	General	General
	EcDsStStagingDiskServer	All Staging Disks owned by the restarted client are released.	All Staging Disks owned by the restarted client are retained, including temporary staging disks.
	EcDsSt8MMServer EcDsStD3Server EcDsStCDROMServer EcDsStPrintServer	N/A	N/A

Some known limitations within the ECS are:

- a.) The Science Data Server accepts asynchronous requests but has only a partial warm start (Landsat 7 requests handled only until 5B).
- b.) Data Distribution may be failing and suspending requests it should not fail (but rather, suspend) or should not suspend (but rather, retry).
- c.) Requests with many sub-requests may experience timing problems because of nested retries or because one of the requests is suspended.

- d.) Coding errors may cause unanticipated fault behavior that is different from what is described above (and such occurrences should be reported as NCR).
- e.) System engineers and designers may have made mistakes in classifying errors, e.g., as fatal versus retry.
- f.) Not all ECS applications use the error recovery capabilities of the ECS Sybase interface infrastructure.

This page intentionally left blank.

5. Limitations of Current Implementation

5.1 Data Server Subsystem

Science Data Server (SDSRV) CSCI

- **Operator GUI:** There is no support for re-installation of ESDTs from the GUI. Reinstallation of ESDTs is supported through a command line Unix Shell script interface. Corresponding data must be manually deleted from the Subscription Server, Advertising and Data Dictionary.
- Metadata update services only support QA metadata.
- There is no persistence of Client requests, except for asynchronous acquire requests. Requests must be re-submitted.
- There is no support for Access Control List checking.
- Certain queries containing constraints against a plural spatial data type (Gpolygon for example) and another constraints against a multi-value attribute (Additional Attributes for example) cannot be performed due to an implementation limitation within the SQS COTS product.

5.2 Client Subsystem

Workbench Software CSCI

- The DAR Tool requires the use of DCE to communicate to the DAR Communications Gateway. As a result, users must have DCE installed on their system. Their workstation must be configured in the DCE cell in order for it to communicate with the DAR Communications Gateway.

5.3 Planning Subsystem

Production Planning CSCI

- The Production Request Editor has not been optimized when creating Production Requests. The creation of large production requests with many input and output granules can take some time. This is mostly due to multiple database accesses; some database accesses may need to be replaced in the future with stored procedures to improve performance.
- The Production Planning Workbench has not been optimized when creating Production Plans. The creation of plans with many data processing requests may take some time. In addition to the database accesses that slow down the Production Request Editor, the Production Planning Workbench's scheduling algorithm may need to be optimized.

- There is no inter-DAAC planning at this time.

5.4 Infrastructure Subsystem

ASTER DAR Gateway CSCI

- The ASTER DAR Client or end users must have valid DCE login in order to communicate with the ASTER DAR Gateway.
- The user must be authorized to perform any of five ASTER DAR Gateway functions including submitDAR, modifyDAR, queryxARContents, queryxARSummary, and queryxARScenes.
- The gateway itself does not extend the DAR functionality; but is limited by the functionality provided by the ASTER GDS through the API set.

E-mail Parser Gateway CSC

- E-mail Parser Gateway only handles Expedited Data Request.
- E-mail Parser Gateway only handles FTTPUSH for media type.

Landsat7 Gateway CSC

- The authentication is limited to comparing the login name and password with the ones stored in the configuration file.
- Landsat7 Gateway does not have a restriction on the number of threads that are spawned, meaning in theory there could be too many threads running at any given time, but in practice this is unlikely to happen.
- Landsat7 Gateway doesn't provide any queuing mechanism. It doesn't have any intelligence. Its functionality is to PASS information between Landsat7 system and the ECS ingest.

Subscription Server CSC/Subscription GUI

- The user can only subscribe to future granules.
- The Subscription Server validates only String type qualifiers.
- There is no persistence for trigger request, subscription request, event registration request, event and subscription update and delete. These requests may be lost or partially finished if some system (including DCE) problems happen when they are processed.
- Support for trigger persistence has not been merged into the current baseline.
- E-mail contents are not clear.
- Subscription Server does not check security issues for updating its database tables. It assumes the client application does this.
- The Operator GUI cannot register or update events on the server side.

- The Operator GUI takes 10 to 15 minutes to come up.
- The Operator GUI on-line help has been deferred until Release 6A.

5.5 MSS Subsystem

MCI CSCI - Security CSC

- If an unauthorized user gains access to a host despite security measures, it is not detected until the next detection interval expires, since this is only checked periodically. (If the detection intervals were decreased, the system uses too much of its processing power for monitoring itself.)
- The security implementation requires the operator to perform security tasks such as running SATAN and running Crack manually.
- The configuration files for TCP Wrappers can become difficult to manage when multiple versions exist as they surely do. If administrators setup a “back door,” the system security can be more easily compromised.

MCI CSCI - Accountability CSC

- There is no retry in place for Database updates or inserts and errors are logged as low priority. Currently, Accountability does not attempt to reconnect to the Database once the connection is lost. Accountability must be restarted to re-establish the connection.

MCI CSCI - Trouble Ticketing CSC

- Trouble Tickets that are forwarded from a DAAC to the SMC are set to a forwarded state in the DAAC. A manual process is necessary to receive notification that the Trouble Ticket has been closed at the SMC and will now be closed at the DAAC.
- While the Trouble Ticket is being worked at the SMC, the M&O operator at the DAAC has little insight as to the current status of the Trouble Ticket.
- For an overall status of Trouble Tickets in the ECS system, reports must be run at each DAAC and forwarded via e-mail where they can be consolidated.

MCI CSCI - Network and Enterprise Management CSC

- If the HP OpenView processes go down, all network hardware and custom software monitoring is incapacitated.
- If the TMR server goes down, all host and COTS software monitoring is incapacitated.
- A single log file adapter monitors all log files and the entry is passed sequentially through all filters that are configured on a particular host. Thus, a generic string such as ERROR would not be of any use in multiple log file configurations. The event would be generated using the first matching configuration and possibly be reported as an event occurring with the incorrect log file.

- Due to an unacknowledged bug in the Tivoli V3.1 log file adapters, running these adapters causes the syslog to crash. Until Tivoli resolves this problem, Tivoli log file adapters are going to be turned off (per an engineering directive) and there is going to be no Tivoli monitoring of COTS logs. An upgrade to Tivoli, V3.6 currently scheduled to occur before Release 5B, can fix this problem.

MCI CSCI - Network and Management CSC

- Network and Management component: ESSM is no longer being supported by Platinum Technologies. ECS will need to migrate to a new product by Platinum called DBVISION. This product has similar functionality to ESSM and is provided as part of Drop 5A.

MLCI CSCI - Baseline Manager CSC

- Control item identifiers - XRP-II uses centralized database technology and is separately installed at each ECS site. This necessitates a special scheme for assigning identifiers to control items so each site may safely exchange database records. For example, the SMC must be able to distribute centrally maintained release records to multiple sites without interfering with records the sites locally maintain there. Similarly, the SMC must be able to absorb copies of site-maintained records to form the consolidated picture of system-wide baselines without contaminating centrally maintained data. To distinguish between centrally maintained and site-maintained records, Baseline Manager expects that identifiers of site-maintained control items have a site 3-character prefix.
- Data entry screens offer form and table views for browsing and editing data records. Table view driver programs cannot handle the number and size of fields used in the form view of numerous screens. Where limitations exist, fields that appear in table view were chosen either because they are best suited to identifying and classifying control items or because they are likely to be used in multi-record operations.
- Import file directories - XRP-II uses the name contained in the IMPORTPATH environment variable as a destination when exporting data records to other sites using the FTP service. Consequently, the directory used to receive the data should have the same name at each site.