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 diagrams and Table 4.8-1 provides descriptions of the interface events shown in the CSS context diagrams. **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.

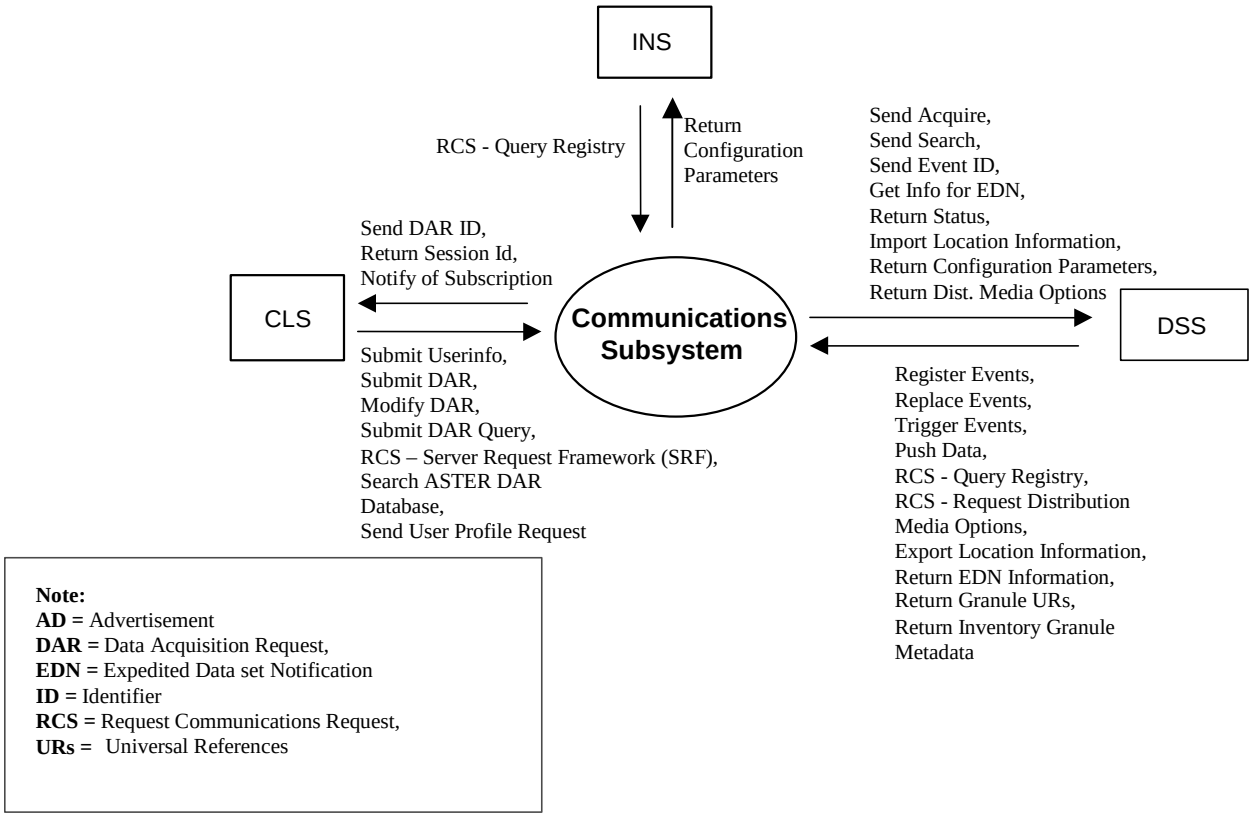


Figure 4.8-1. Communications Subsystem (CSS) Context Diagram

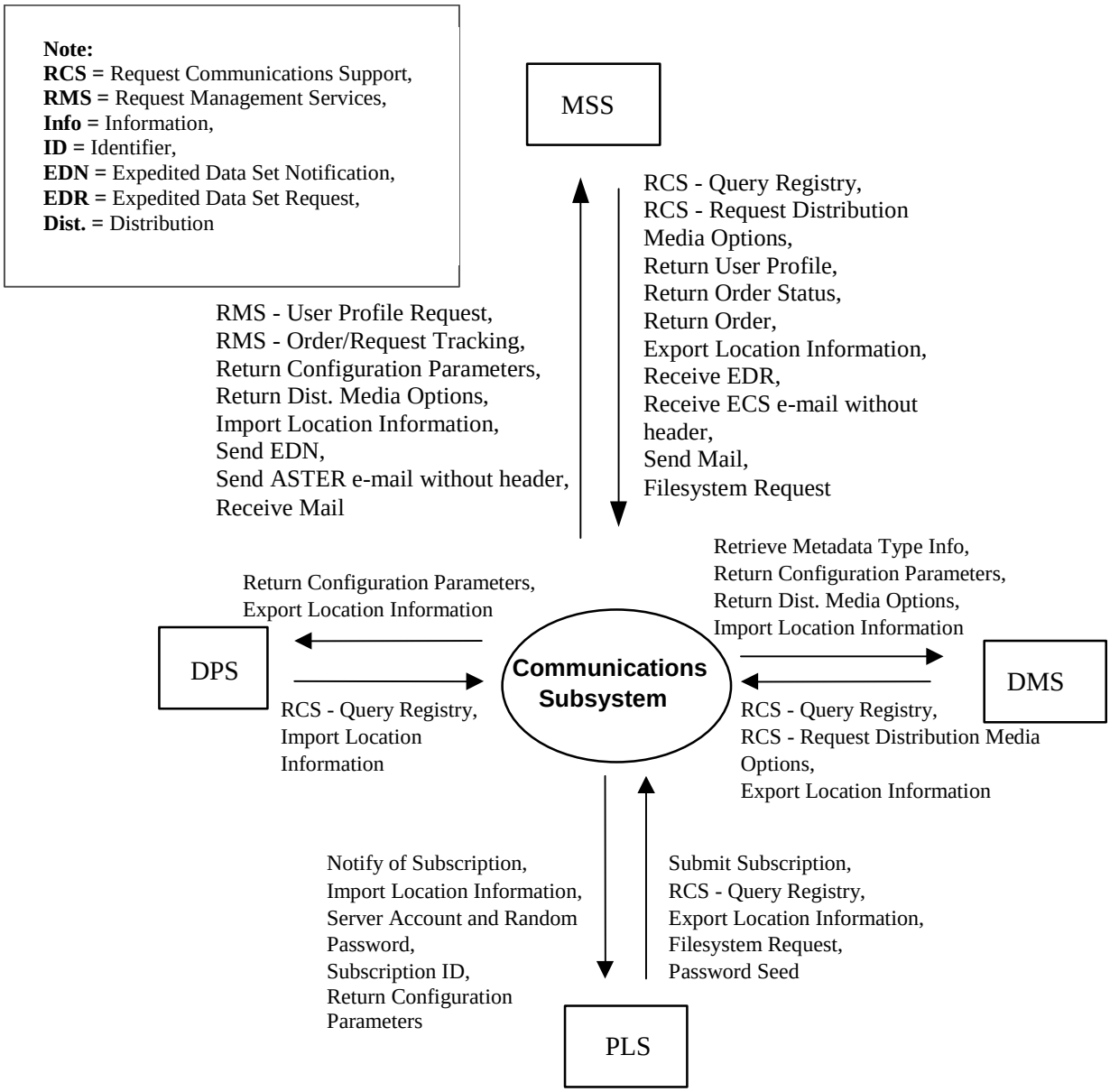


Figure 4.8-1. Communications Subsystem (CSS) Context Diagram (cont.)

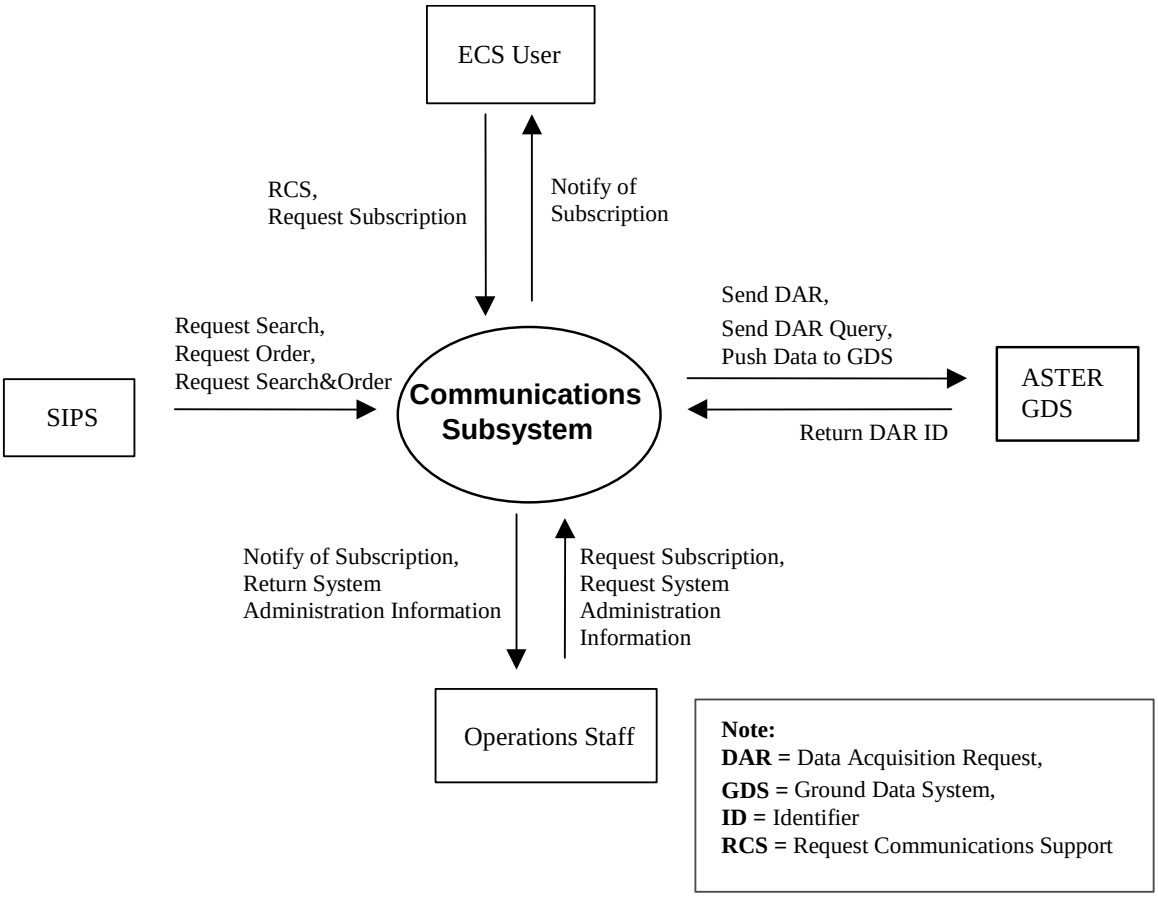


Figure 4.8-1. Communications Subsystem (CSS) Context Diagram (cont.)

Table 4.8-1. Communications Subsystem (CSS) Interface Events (1 of 5)

Event	Interface Event Description
Send Acquire	An “acquire” (instruction to obtain data) is created by the CSS and sent to the **DSS** via CCS Middleware calls. 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 Search	The CSS sends search requests received via the SIPS interface to the **DSS** on behalf of an external ECS user.
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.

Table 4.8-1. Communications Subsystem (CSS) Interface Events (2 of 5)

Event	Interface Event Description
Return Status	Status returned by the CSS to the **DSS** to simply indicate that the request was received, not that the action succeeded.
Import Location Information	The **DSS** retrieves physical and logical server location information from the CSS.
Return Configuration Parameters	The **INS, DSS, MSS, DMS, PLS** and **DPS** receive the configuration parameters and associated values from the Registry Server within the CSS.
Return Dist. Media Options	The **DSS** receives the requested distribution media options from the CSS.
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 subsystem services required to perform specific assignments are requested from the CSS. These services include: • CCS Middleware Support • Database Connection Services • 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 • Mode Information • Query Registry - Retrieving the requested configuration attribute-value pairs from the Configuration Registry • Request Distribution Media Options from the Configuration Registry
Export Location Information	The **DSS**, **DMS**, **MSS** and **PLS** send physical and logical server location information to the CSS for data location.
Return EDN Information	The CSS receives and uses the Expedited Data Set Notification (EDN) information from the **DSS** to send messages to users at the ASTER Ground Data System (GDS).

Table 4.8-1. Communications Subsystem (CSS) Interface Events (3 of 5)

Event	Interface Event Description
Return Granule URs	The CSS receives Earth Science Data Type (ESDT) Universal References (URs) for the requested granules from the **DSS**.
Return Inventory Granule Metadata	The CSS (MTMGW Server) receives the inventory granule metadata identifying the scene within the granule from the **DSS** based on an inventory search request.
Submit Userinfo	The **CLS** sends the user name and password to the CSS. The CSS returns a session id.
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.
Search ASTER DAR Database	The **CLS** submits a request to the CSS to search the ASTER GDS DAR database for DARs and their respective status (i.e., acquired scenes). Search qualifications may be in the form of DAR parameters or DAR Ids. To get a status of the search, users may view the Search Status via the Java DAR Tool.
Send User Profile Request	The **CLS** (via the user) sends a user profile to the CSS to send ASTER requests between the ECS and ASTER GDS.
Send DAR ID	As the result of a successfully submitted DAR, the user receives a DAR ID from the CSS via the **CLS**. This is a string of characters used to track a DAR.
Return Session Id	The **CLS** receives a session id from the CSS to communicate with the ASTER GDS.
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 **CLS, PLS** (Subscription manager or the On-Demand Processing Manager)**, ECS User** or **Operations Staff**.
Return User Profile	The CSS receives user profile information from the **MSS** (Accountability Management Service) to authenticate a user.
Return Order Status	The CSS receives an order id and status for the requested ECS product from the **MSS** (Accountability Management Service).
Return Order	The MSS returns the product order object to the requester via the **CSS.**
Receive EDR	The **MSS** strips the EDR header and sends the EDR to the CSS (E-mail Parser Gateway Server CSC).
Receive ECS e-mail without header	The header is removed from the inbound message (by **MSS**), logged, and forwarded to the predefined ECS recipient of the e-mail alias by the CSS.
Send Mail	The **MSS** uses the CsEmMailRelA interface to send mail to the CSS for distribution to ASTER GDS users.
Filesystem Request	The CSS (NFS client) receives requests for ECS files and directories via an established mount point from the **MSS** and **PLS**. The CSS (NFS Server) makes the storage device(s) and its data accessible for use by the clients.

Table 4.8-1. Communications Subsystem (CSS) Interface Events (4 of 5)

Event	Interface Event Description
Retrieve Metadata Type Info	The CSS retrieves type information for qualifying metadata specified in a SIPS search request from the **DMS**.
Return Dist. Media Options	The CSS returns distribution media options to the **DMS** and **MSS** for distribution requests.
Import Location Information	The **DMS**, **MSS** and **PLS** request server location information from the CSS for data searching purposes.
Submit Subscription	The **PLS** creates a subscription, sent to the CSS, using the advertisement for subscribing to an insert event for an ESDT.
Password Seed	The **PLS** requests an account and provides a password to the CSS.
Server Account and Random Password	The **PLS** receives an account and random password from the CSS after providing a password seed to establish an account.
Subscription ID	The **PLS** receives a subscription identifier from the CSS after submitting a subscription.
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 includes: • **System startup and shutdown** – Please refer to the release-related, current version of the Mission Operations Procedures for the ECS Project document (611) and the current ECS Project Training Material document (625), identified in Section 2.2.1 of this document. • **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. • **Order/Request Tracking** – The **MSS** provides an order tracking service to requesting subsystems to create and track user product orders by request.
Send EDN	The CSS sends the EDN to the **MSS** to forward to the ASTER GDS.
Send ASTER e-mail without header	The CSS sends an e-mail message to a predefined ASTER e-mail alias within the ECS (in **MSS**), without a header (e.g., Expedited Data Set Notification or EDN).
Receive Mail	The CSS returns mail to the **MSS** from ASTER GDS users to be distributed to addressees.
Send DAR	The CSS sends the DAR to the **ASTER GDS** Storage Server.
Send DAR Query	The CSS sends the DAR query to the **ASTER GDS** DAR Server.
Push Data to GDS	The CSS provides the communications infrastructure to push data from the DSS to the **ASTER GDS**.
Return DAR ID	The **ASTER GDS** returns a DAR ID to the CSS (ASTER DAR Gateway Server) at the ECS.
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.

Table 4.8-1. Communications Subsystem (CSS) Interface Events (5 of 5)

Event	Interface Event Description
Request System Administration Information	The **Operations Staff** requests information on system administration including application administration, fault metrics, performance metrics and system alarms.
Return System Administration Information	The **Operations Staff** receives information on system administration including application administration, fault metrics, performance metrics and system alarms.
Request Search	Search requests are sent to the CSS via the **SIPS** interface.
Request Order	Order requests are sent to the CSS via the **SIPS** interface.
Request Search&Order	Integrated search and order requests are sent to the CSS via the **SIPS** interface.
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 includes: • **Order/Request Tracking** – The **MSS** provides an order tracking service to requesting subsystems to create and track user product orders by request.

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

Note: The following RogueWave Libraries are currently delivered with custom code as static libraries. A separate installation of dynamic libraries is no longer required.

- RogueWave's Tools.h++

  The Tools.h++ class libraries provide basic functions and objects such as strings and collections.

- RogueWave's DBTools.h++

  The DBTools.h++ C++ class libraries provide interaction, in an object-oriented manner, to the Sybase ASE database SQL server. The DBTools provide a buffer between the CSS processes and the relational database used.

- RogueWave's Net.h++

  The Net.h++ C++ class libraries, which provide functions and templates that facilitate writing applications, which communicate with other applications.

Other COTS products include:

- 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 Adaptive Server Enterprise (ASE)

  The Sybase ASE provides access for the Subscription Server to insert, update and delete Subscription Server database information. The Sybase ASE 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 ASE.

- CCS Middleware

  CCS Middleware provides a common NameServer Mechanism, which packages the common portions of the communication mechanisms into global objects to be used by all subsystems. It provides a set of standard CCS Proxy/Server classes, which encapsulates all of the common code for middleware communications (e.g., Portals, Couplers, RWCollectables, etc.). It also provides a code generator, which produces the application specific proxy & server code. This allows the software engineer to concentrate on the application specific code without worrying about the infrastructure.

- UNIX Network Services

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

Error Handling and processing

EcUtStatus is a class used throughout the ECS custom code for general error reporting. It is almost always used as a return value for functions and allows detailed error codes to be passed back up function stacks.

When an error occurs, the error is logged into the applications log (ALOG). The Communications Subsystem (CSS) and System Management Subsystem (MSS) have two main mechanisms to handle the error:

1. Return an error status
2. Throw an exception.

The CSS uses the following classes for error handling and processing:

The EcUtStatus class is used to describe the operational status of many functions. The values most often reported are "failed" and "ok." But depending upon the application, detailed values could be set and sent. Please refer to the definition of this class (located in /ecs/formal/COMMON/CSCI_Util/src/Logging/EcUtStatus.h) for all possible values.

The EcPoError class defines the basic error types and handling functions for using the EcPoConnectionsRW class (based upon RWDBTool++). The CSS Subscription Server, IOS Server and MSS Order tracking Server use the EcPoConnectionsRW class.

The RWCString is used to store some status value returned by applications.

Integer is used to return some error status by applications. This is used specifically between client and server communications.

Many types of exceptions can be sent and handled by the CSS. These include exceptions sent by Commercial Off The Shelf (COTS) products (such as DCE, RWDBTools++, RWTools++), systems and exceptions defined by individual applications.

For writing messages to the debug log, the following macros are used:

PF_STATUS writes a message at a "log level" of 1 to the debug log. For example, PF_STATUS

```
{
cerr << "EcSbSubServer: Host name = " << hostInfo.nodename << endl;
}
```

PF_VERBOSE writes a message at a "log level" of 2 to the debug log. For example, PF_VERBOSE

```
{
cerr << "EcSbUpdateSubRequest:Update: Succeeded." << endl;
}
```

PF_DEBUG writes a message at a "log level" of 3 to the debug log. For example, PF_DEBUG

```
{
cerr << "EcSbSubscription:Execute: Action = ACQUIRE : " << endl;
cerr << *tmpAction << endl;
}
```

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 /ecs/formal/CSS/DOF/src/SUBSCRIPTION) provides the capability to accept subscriptions, and process subscriptions upon event notification. Events are made available to users through the during the user registration process. Subscriptions are submitted events. 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, 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 /ecs/formal/CSS/DOF/src/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 database 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 /ecs/formal/CSS/DOF/src/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 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 stages the EDS granule metadata and passes the request on to the Distribution Server for distribution to the client.

4. The MOJO Gateway Server (the software is found under /ecs/formal/CSS/DOF/src/RELB_GATEWAY/MOJO) provides a common interface and network address for the following distributed ECS services: User profiles, 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. Subscriptions are placed with the CSS Subscription Server.

5. The Configuration Registry Server (the software is found in directory /ecs/formal/CSS/DOF/src/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.

6. The Machine-to-Machine Gateway (the software is found in directory /ecs/formal/CSS/DOF/src/RELB_GATEWAY/MTMGW) The Machine-to-Machine Gateway (MTMGW) Server provides an automated ordering capability to allow the Science Investigator-Led Processing Systems (SIPS) to reprocess data externally from the ECS. In order to safeguard communications between the MTMGW Server and the SIPS, a SSH (Secure Shell protocol) is employed to secure the line.

7. The CCS Name service group is a COTS software set of Name and Time Services.

   - The ECS Naming Service (the software is found in directory /ecs/formal/CSS/DOF/src/NS/naming) 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 ECS Naming Service, independent of physical location. The clients use the ECS Naming Service to find servers based on an operating mode. This is the primary way clients locate servers.

   - The Time Service (the software is found in directory /ecs/formal/CSS/DOF/src/TIME/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 /ecs/formal/CSS/DOF/src/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 INGST CSCI uses the FTP application to ingest data from external data providers. The STMGT CSCI uses the FTP application to electronically distribute data to users.

9. FTP Notification (the software is found in directory /ecs/formal/CSS/DOF/src/FTP) provides successful completion notifications for FTP (get) data pulls and (put) data pushes. The INGST CSCI provides notifications to external data providers of data ingest into the ECS and the DDIST/PDSIS CSCIs provide notifications for data distribution from the ECS.

10. BDSpro (no physical directory) is a non-standard extension to the Network File System (NFS) that handles large file transfers on SGI IRIX platforms. BDSpro is a fast file transfer utility to move large data files over high-speed networks such as the high-speed Gig Ethernet communications lines. BDSpro exploits the data access speed of the NFS file system and data transfer rates of network media, such as high-speed Gig Ethernet, 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 STMGT CSCI uses BDSpro to archive data produced by the Data Processing Subsystem.

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 Metadata Configuration Files (MCFs) to the DDIST CSCI staging disks.

13. The mail support group provides electronic mail service.

    - E-mail (the software is found in directory/ecs/formal/CSS/DOF/src/EMAIL/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 /ecs/formal/CSS/DOF/src/AUTHN) provides processes a means for obtaining random passwords and gaining access to Sybase.

16. The Domain Name Service (DNS, the software is found in directory ecs/formal/CSS/NameServer/src) 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 4.8-33 for an illustration of the domain tree hierarchy.

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

    - Process Framework (PF) (the software is located in directory /ecs/formal/CSS/DOF/src/PF/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 (CCS Middleware naming service), 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 CCS Middleware 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/src/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 CCS Middleware calls, message passing and persistent storage as a CSS support capability with the described features available by subsystem request. SRF uses CCS Middleware calls.

- Message Passing (the software is found in directory /ecs/formal/CSS/DOF/src/MP-OODCE_NO1): Message Passing provides peer-to-peer asynchronous communications services 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/src/UR/framework): Universal References (URs) provide applications and users a system wide mechanism for referencing ECS data and service objects. Manipulating logical entities represented at run time as C++ objects in virtual memory performs ECS functions. Users and applications require references to the logical entities beyond the effective computational time to keep the objects in memory. Therefore, applications and users are given URs to these objects. Once an

UR is made for an object, the object can be disposed of and later reconstituted from the UR. URs take up a small fraction of the space to keep in memory and can be externalized into an ASCII string, which an end user can manage. URs have the capability of re-accessing and/or reconstituting the object into memory as needed. Therefore, the object does not have to remain in memory, and can if appropriate, be written to a secondary storage system, like a database. While the UR mechanism guarantees reliable data externalization and internalization, the content of each type of UR is application specific. Only the object (this is referred to as the "UR Provider") that initially provides the UR is allowed to access and understand its content. URs are strongly typed to enforce appropriate access control to internal data both at compile time and during run time. Since URs are typed and have object specific data in them, separate UR object classes exist for each UR Provider class referred to. All of these UR classes use the mechanisms provided by the UR framework.

- Event Logging (the software is found in directoryLOGGING): 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 /ecs/formal/CSS/DOF/src/NS/service_locator): 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 CCS Middleware-based communication.
- Failure Recovery Framework (the software is found in directory /ecs/formal/CSS/DOF/src/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 CCS Naming Service, 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.
- Time Service (the software is found in directory /ecs/formal/CSS/DOF/src/TIME/time): the class providing the structured time information and get RogueWave type of time information.

- 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.

4.8.1.1 Subscription Server Computer Software Component Description

4.8.1.1.1 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. 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.

4.8.1.1.2 Subscription Server Context

Figure 4.8-2 is the Subscription Server context diagram. Table 4.8-2 provides descriptions of the interface events in the Subscription Server context diagram.

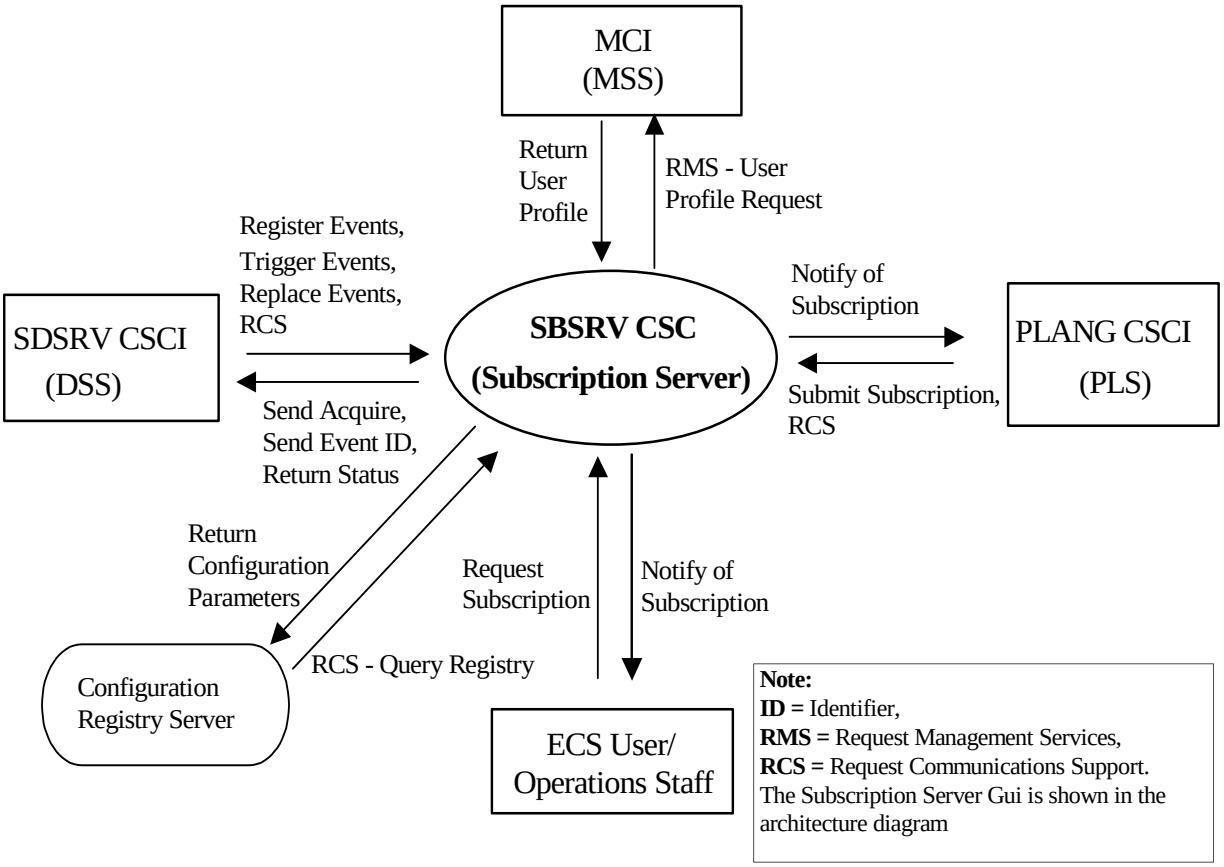


Figure 4.8-2. Subscription Server Context Diagram

Table 4.8-2. Subscription Server Interface Events (1 of 2)

Event	Interface Event Description
Request Management Services	The **MCI** 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: • **System startup and shutdown -** Please refer to the release-related, current version of the Mission Operations Procedures for the ECS Project document (611) and the current ECS Project Training Material document (625), identified in Section 2.2.1 of this document. • **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.
Notify of Subscription	The SBSRV CSC sends notification (E-mail or inter-process) to the **ECS User**, **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** and **CSMS** CSCI/CSC. The CSCI services required to perform specific assignments are requested from the DCCI CSCI. These services include: • CCS Middleware Support • Database Connection Services • 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 • Mode Information • Query Registry - Retrieving the requested configuration attribute-value pairs from the Configuration Registry • Request Distribution Media Options from the Configuration Registry
Request Subscription	A subscriber (**ECS user** or **Operations Staff** - optionally) sends information with the subscription, specifying an action(s) (e.g., acquire or update) to be taken when the subscribed event occurs.
Return Configuration Parameters	The **Configuration Registry CSC** returns the requested configuration parameters to the Subscription Server (SBSRV) CSC.
Send Acquire	An “acquire” (instructions to obtain data) is created by the DCCI CSCI and sent to the **SDSRV CSCI** via CCS Middleware calls. 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.

Table 4.8-2. Subscription Server Interface Events (2 of 2)

Event	Interface Event Description
Return status	Status returned by the DCCI CSCI (SBSRV CSC) 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 (ESDT) to the DCCI CSCI (SBSRV CSC) 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 ESDT to the SBSRV CSC (within the DCCI CSCI) when an ESDT is updated. This event replaces the original event in the SDSRV CSCI.
Return User Profile	The **MCI** returns the user profile to the subscription server for user authentication for subscriptions.

4.8.1.1.3 Subscription Server Architecture

Figure 4.8-3 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.

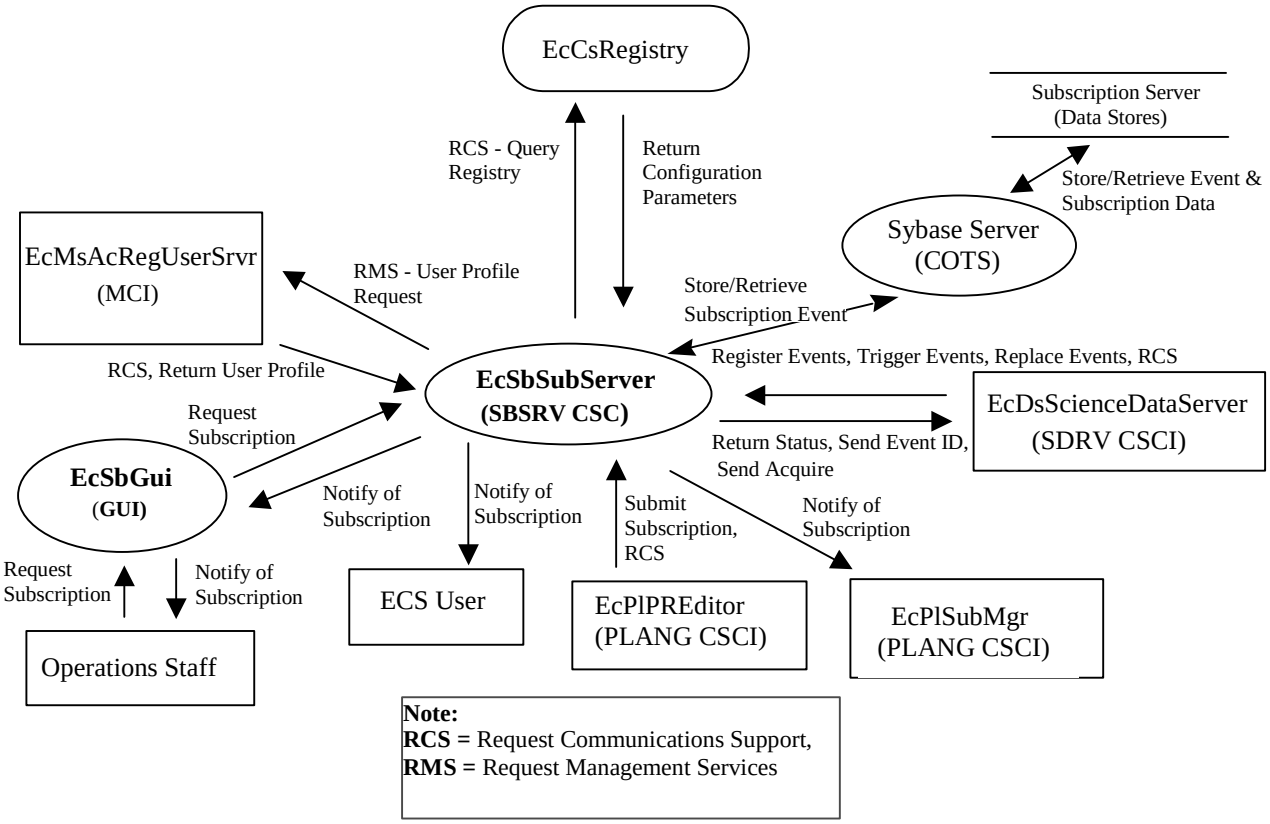


Figure 4.8-3. Subscription Server Architecture Diagram

4.8.1.1.4 Subscription Server Process Descriptions

Table 4.8-3 provides descriptions of the processes shown in the Subscription Server architecture diagram.

Table 4.8-3. Subscription Server Processes

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

4.8.1.1.5 Subscription Server Process Interface Descriptions

Table 4.8-4 provides descriptions of the interface events shown in the Subscription Server architecture diagram.

Table 4.8-4. Subscription Server Process Interface Events (1 of 5)

Event	Event Frequency	Interface	Initiated by	Event Description
Request Communications Support	One service per request	***Process:*** EcCsIdNameServer ***Libraries:*** EcPf, Middleware, FoNs, FoIp, oodce *Classes:* EcPfManagedServer, CCSMdwNameServer, FoNsNameServerProx y, CCSMdwRwNetProxy ***Library (Common):*** EcUr *Class:* EcUrServerUR ***Library:*** event *Class:* EcLgErrorMsg ***Process:*** EcSbSubServer *Library:* EcSbCl *Classes:* EcClEvent, EcClTriggerEventCb, EcClRegisterEventCb ***Process:*** EcCsRegistry *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	***Process:*** EcDsScienceDataServer *Classes:* DsDeEventCustomizer, DsBtSbrvNotifier ***Process:*** EcPlPREditor *Library:* PlCore1 *Classes:* Most PLS Classes ***Process:*** EcMsAcRegUserSrvr *Libraries:* MsAcClnt, MsAcComm *Class:* MsAcUsrProfile	The DCCI CSCI provides a library of services available to each SDPS and CSMS CSCI/CSC. The CSCI services required to perform specific assignments are requested from the DCCI CSCI. These services include: • CCS Middleware Support • Database Connection Services • File Transfer Services • Network & Distributed File Services • Bulk Data Transfer Services • Name/Address Services • Password Services • Server Request Framework (SRF) • Universal Reference (UR) • Error/Event Logging • Message Passing • Fault Handling Services • Mode Information • Query Registry - Retrieving the requested configuration attribute-value pairs from the Configuration Registry • Request Distribution Media Options from the Configuration Registry

Table 4.8-4. Subscription Server Process Interface Events (2 of 5)

Event	Event Frequency	Interface	Initiated by	Event Description
Return Configuration Parameters	One per configuration registry query	*Process:* EcSbSubServer *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	*Process:* EcCsRegistry	The EcCsRegistry returns the requested configuration parameters to the EcCsMojoGateway.
Store/Retrieve Subscription Event	One per store and retrieve event	Sybase ASE (COTS)	*Process:* EcSbSubServer *Library:* EcSbSr *Classes:* EcSbEventStore, EcSbSubscriptionStore	The EcSbSubServer stores and retrieves subscription information and events from the Subscription Server Data Stores via the **Sybase ASE**.
Store/Retrieve Event & Subscription Data	One per request	Sybase ASE (COTS)	*Process:* EcSbSubServer *Library:* EcSbSr *Classes:* EcSbEventStore, EcSbSubscriptionStore	The EcSbSubServer stores and retrieves event and subscription data via the **Sybase ASE** 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	*Process:* EcSbSubServer *Library:* EcSbCl *Classes:* EcClEvent, EcClRegisterEventCb	*Process:* EcDsScienceDataServer *Library:* DsDe1sh *Class:* 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	*Process:* EcSbSubServer *Library:* EcSbCl *Classes:* EcClEvent, EcClTriggerEventCb	*Process:* EcDsScienceDataServer *Library:* DsBtSh *Class:* DsBtSbsrvNotifier	The **EcDsScienceDataServer** notifies the EcSbSubServer (via an event trigger) when a subscription event occurs on an Earth Science Data Type Service.

Table 4.8-4. Subscription Server Process Interface Events (3 of 5)

Event	Event Frequency	Interface	Initiated by	Event Description
Replace Events	One per ESDT update	*Process:* EcSbSubServer *Library:* EcSbSrSh *Class:* EcClEvent	*Process:* EcDsScienceDataServer *Class:* 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	*Process:* EcDsScienceDataServer *Library:* DsBtSh *Class:* DsBtSbsrvNotifier	*Process:* EcSbSubServer *Library:* 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	*Process:* EcDsScienceDataServer *Library*: DsDe1Sh *Class*: DsDeDataDictController	*Process:* EcSbSubServer *Library* EcSbCl *Class:* EcClEvent	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	*Process:* EcDsScienceDataServer *Library:* DsCl *Classes:* DsClRequest, DsClCommand, DsClESDTReferenceCollector	*Process:* EcSbSubServer *Library:* EcSbSr *Class:* EcSbSubscription	An “acquire” (instruction to obtain data) is created by the EcSbSubServer and sent to the **EcDsScienceDataServer** via CCS Middleware calls.
Notify of Subscription	One per subscription submitted	***Process:*** EcPlSubMgr *Class:* PlSubMsgCb ***Process:*** Email Daemon Classes: CsEmMailRelA, EcCsNotify.cxx	*Process:* EcSbSubServer *Library:* EcSbSr *Classes:* EcSbSubscription, EcSbNotification	The EcSbSubServer sends E-mail to the **ECS User**, **Operations Staff** (via the Email class), or inter-process notification (via the message-passing framework) to the **EcPlSubMgr**.

Table 4.8-4. Subscription Server Process Interface Events (4 of 5)

Event	Event Frequency	Interface	Initiated by	Event Description
Submit Subscription	One per subscription with acquire request	*Process:* EcSbSubServer *Library:* EcSbCl *Class:* EcClSubscription	*Process:* EcPlPREditor_IF *Library:* PlCore1 *Class:* PlDataType	The **EcPlPREditor_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	*Process:* EcSbSubServer *Library:* EcSbSr *Classes*: EcSbSubmitSubRequest, EcSbSubscription	Operations Staff *Process:* EcSbGui *Library:* EcSbCl *Class:* EcClSubscription	The **Operations Staff** can make a request for a subscription to the EcSbSubServer via the EcSbGui on behalf of an ECS User.
Return User Profile	One per profile request	*Process:* EcSbSubServer *Class:* EcSbSr	*Process:* EcMsAcRegUserSrvr *Libraries:* MsAcClnt, MsAcComm *Class:* EcAcProfileMgr	The EcSbSubServer receives user profile information from the **EcMsAcRegUserSrvr** to authenticate a user.

Table 4.8-4. Subscription Server Process Interface Events (5 of 5)

Event	Event Frequency	Interface	Initiated by	Event Description
Request Management Services (RMS)	One per service request	N/A	N/A	The **EcMsAcRegUserSrvr** provides 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 the items listed below.
RMS (cont.)	At system startup or shutdown and for restarts	*Process:* EcSbSubServer	DAAC unique startup scripts	**System startup and shutdown -** Please refer to the release-related, current version of the Mission Operations Procedures for the ECS Project document (611) and the current ECS Project Training Material document (625), identified in Section 2.2.1 of this document.
RMS (cont.)	One per request	*Process:* EcMsAcRegUserSrvr *Libraries:* MsAcClnt, MsAcComm *Class:* EcAcProfileMgr	*Process:* EcSbSubServer *Class:* EcSbSr	**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.

4.8.1.1.6 Subscription Server Data Stores

Subscription Server uses the COTS software Sybase ASE 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 4.8-5 provides descriptions of the data found in the seven separate Sybase ASE 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, 311-CD-626).

Table 4.8-5. Subscription Server Data Stores

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.
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.

4.8.1.2 ASTER DAR Gateway Server Software Description

4.8.1.2.1 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)).

4.8.1.2.2 ASTER DAR Gateway Server Context

Figure 4.8-4 is the ASTER DAR Gateway Server context diagram and Table 4.8-6 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.

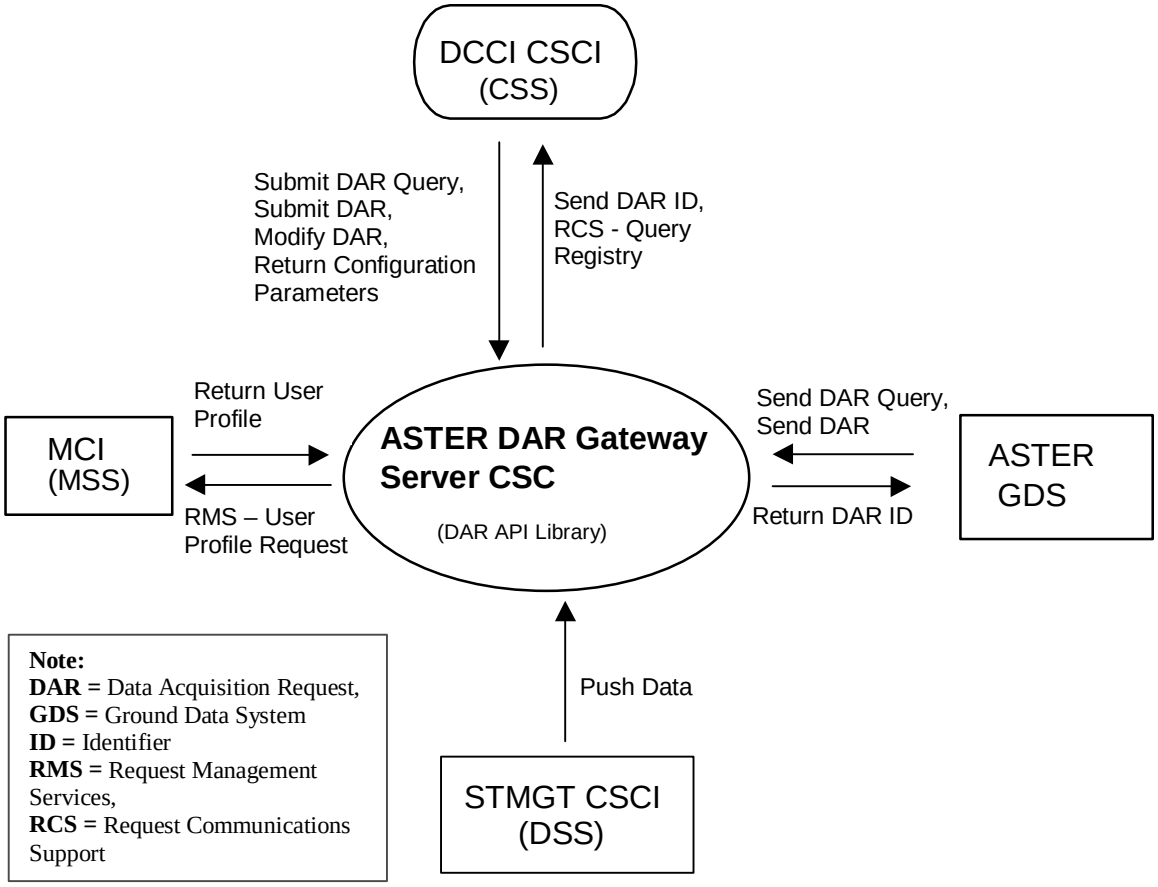


Figure 4.8-4. ASTER DAR Gateway Server Context Diagram

Table 4.8-6. ASTER DAR Gateway Server Interface Events (1 of 2)

Event	Interface Event Description
Send DAR ID	The ASTER DAR Gateway Server extracts the returned DAR ID and sends it to the Java DAR Tool, via the **DCCI CSCI**.
Request Communications Support (RCS)	The DCCI CSCI provides a library of services available to **each SDPS** and **CSMS CSCI/CSC**. The CSCI services required to perform specific assignments are requested from the DCCI CSCI. These services include: • CCS Middleware Support • Database Connection Services • File Transfer Services • Network & Distributed File Services • Bulk Data Transfer Services • Name/Address Services • Password Services • Server Request Framework (SRF) • Universal Reference (UR) • Error/Event Logging • Fault Handling Services • Mode Information • Query Registry - Retrieving the requested configuration attribute-value pairs from the Configuration Registry
Send DAR Query	The ECS user at the **ASTER GDS** sends the DAR query to the ASTER GDS DAR Server by DAR API library calls, which communicates via TCP/IP sockets over the EOSDIS Mission Support Network (EMSn).
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 the EMSn. The ASTER GDS returns a DAR ID to the ASTER DAR Gateway Server CSC at the ECS. The ASTER DAR Gateway Server extracts the returned DAR ID and sends it to the ASTER DAR tool, via the MOJO Gateway Server CSC.
Return DAR ID	The **ASTER GDS** returns a DAR ID to the ASTER DAR Gateway Server at the ECS.
Push Data	The **STMGT CSCI** pushes data (i.e., EDS), via the FTP service and an FTP Daemon, to the ASTER DAR Gateway for data distribution per user request. A signal file is also sent to indicate the completion of the file transfer by particular ESDTs that function this way.
Request Management Services	The **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 include: • **System startup and shutdown -** Please refer to the release-related, current version of the Mission Operations Procedures for the ECS Project document (611) and the current ECS Project Training Material document (625), identified in Section 2.2.1 of this document. • **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 4.8-6. ASTER DAR Gateway Server Interface Events (2 of 2)

Event	Interface Event Description
Return User Profile	The ASTER DAR GTWAY CSC receives user profile information from the **MCI** to authenticate a user.
Submit DAR Query	The **DCCI CSCI** sends a DAR query, received from the JAVA DAR Tool and of type queryxARContents, queryxARSummary, or queryxARScenes, to the ECS ASTER DAR Gateway Server via sockets.
Submit DAR	The **DCCI CSCI** submits DARs, received from the JAVA DAR Tool, to the ECS ASTER DAR Gateway Server via sockets.
Modify DAR	The **DCCI CSCI** sends modified DARs, received from the JAVA DAR Tool, to the ECS ASTER DAR Gateway Server via sockets.
Return Configuration Parameters	The **DCCI CSCI** sends the requested configuration parameters to the ASTER DAR Gateway Server.

4.8.1.2.3 ASTER DAR Gateway Server Architecture

Figure 4.8-5 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.

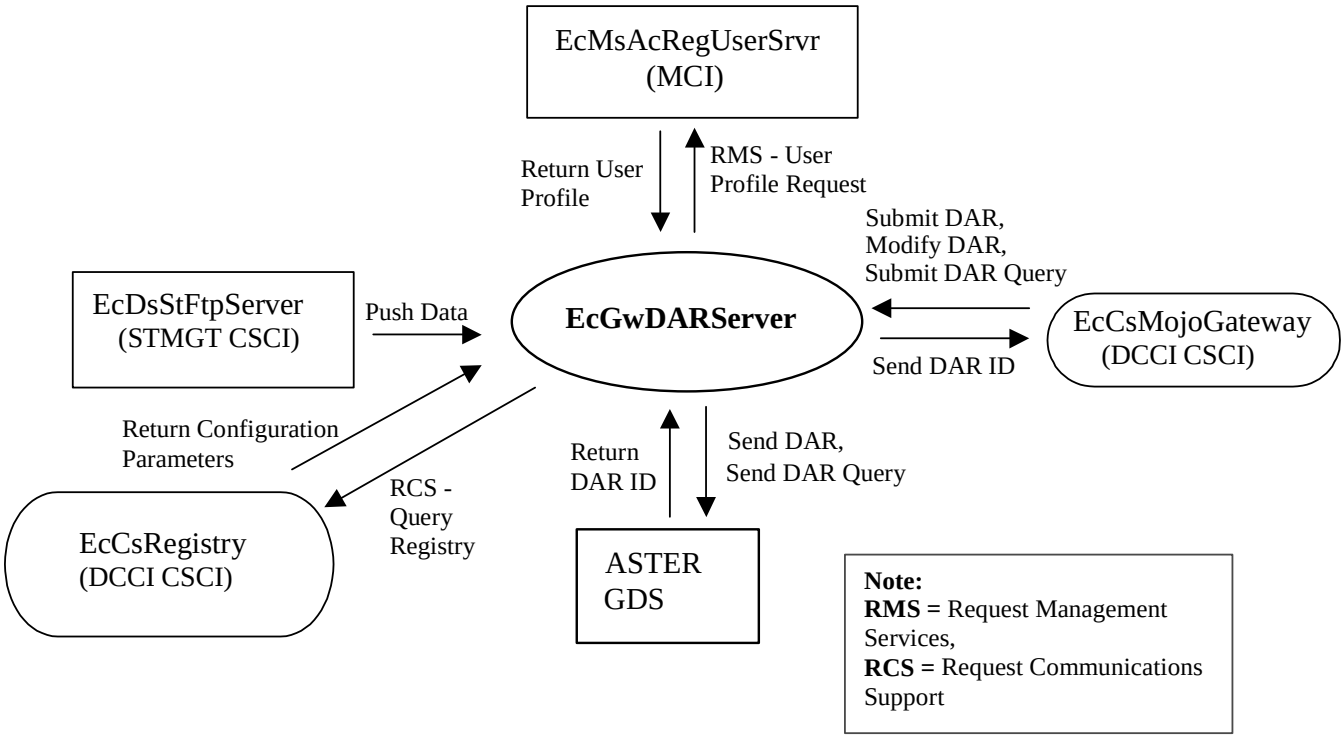


Figure 4.8-5. ASTER DAR Gateway Server Architecture Diagram

4.8.1.2.4 ASTER DAR Gateway Server Process Descriptions

Table 4.8-7 provides descriptions of the processes in the ASTER DAR Gateway Server architecture diagram.

Table 4.8-7. ASTER DAR Gateway Server Processes

Process	Type	COTS/ Developed	Functionality
EcGwDARServer	Server	Developed	The ASTER DAR Gateway Server provides five functions. • 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. Synchronous request processing is supported. Asynchronous request processing is supported. Multiple concurrent requests are supported.

4.8.1.2.5 ASTER DAR Gateway Server Process Interface Descriptions

Table 4.8-8 provides the descriptions of the interface events shown in the ASTER DAR Gateway Server architecture diagram.

Table 4.8-8. ASTER DAR Gateway Server Process Interface Events (1 of 4)

Event	Event Frequency	Interface	Initiated by	Event Description
Request Management Services (RMS)	One per service request	N/A	N/A	The **EcMsAcRegUserSrvr** provides 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 the items listed below.
RMS (Cont.)	At system startup or shutdown and for restarts	*Process:* EcGwDARServer	DAAC unique startup scripts	**System startup and shutdown -** Please refer to the release-related, current version of the Mission Operations Procedures for the ECS Project document (611) and the current ECS Project Training Material document (625), identified in Section 2.2.1 of this document.
RMS (Cont.)	One per user request	*Process:* EcMsAcRegUserSrvr *Libraries:* MsAcClnt, MsAcComm *Class:* EcAcProfileMgr	*Process:* EcGwDARServer *Class:* EcGwDARGatewayRequest_S	**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.
Submit DAR	One per DAR	*Process:* EcGwDARServer *Library:* EcGwDAR *Class:* EcGwDARSubmitDarRequest_C	User *Process:* EcCsMojoGateway *Class*: EcMjDarSubmitDarProxy	A user submits a DAR, through the **EcCsMojoGateway**, to the ECS EcGwDARServer via sockets.

Table 4.8-8. ASTER DAR Gateway Server Process Interface Events (2 of 4)

Event	Event Frequency	Interface	Initiated by	Event Description
Modify DAR	One per modified DAR	*Process:* EcGwDARServer *Library:* EcGwDAR *Class:* EcGwDARModifyDarRequest_C	*Process:* EcCsMojoGateway *Class*: EcMjDarModifyDarProxy	A user modifies a DAR and sends the modified request, through the **EcCsMojoGateway**, to the ECS EcGwDARServer via sockets.
Submit DAR Query	One per query request	*Process:* EcGwDARServer *Library:* EcGwDAR *Classes:* EcGwDARQueryxARContentsRequest_C, EcGwDARQueryxARSummaryRequest_C EcGwDARQueryxARScenesRequest_C	*Process:* EcCsMojoGateway *Classes*: EcMjDarQueryxARContentsProxy, EcMjDarQueryxARSummaryProxy, EcMjDarQueryxARScenesProxy	The **EcCsMojoGateway** sends the query request, received from the EcClWbJdt (JAVA DAR Tool), to the ECS ASTER DAR Gateway Server via sockets.
Send DAR ID	One per DAR ID sent	*Process:* EcCsMojoGateway *Library:* EcGwDAR Class: EcGwDARSubmitDarRequest_C	*Process:* EcGwDARServer *Class:* EcGwDARGatewayRequest_S	The EcGwDARServer extracts the returned DAR ID and sends it to the EcClWbJdt (Java DAR Tool), via the EcCsMojoGateway.
Send DAR	One per ASTER DAR send	ASTER GDS	*Process*: EcGwDARServer *Library:* IcDarApi *Class*: EcGwDARGatewayRequest_S	The EcGwDARServer sends the request to the **ASTER GDS** Storage Server via the DAR API library, which communicates via TCP/IP sockets over theEOSDIS Mission Support network (EMSn). 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 EcClWbJdt (Java DAR tool), via the EcCsMojoGateway.

Table 4.8-8. ASTER DAR Gateway Server Process Interface Events (3 of 4)

Event	Event Frequency	Interface	Initiated by	Event Description
Send DAR Query	One per query request	ASTER GDS	*Process:* EcGwDARServer *Library:* IcDarApi *Class:* EcGwDARGatewayRequest_S	The EcGwDARServer sends the query to the **ASTER GDS** DAR Server via DAR API Library calls, which communicates via TCP/IP sockets over the EMSn.
Return DAR ID	One per DAR ID return	*Process:* EcGwDARServer *Library:* IcDarApi	ASTER GDS	The **ASTER GDS** returns a DAR ID to the EcGwDARServer at the ECS.
Request Communications Support (RCS)	One service per request	***Process:*** EcCsIdNameServer ***Libraries:*** EcPf, Middleware, FoNs, FoIp, oodce *Classes:* EcPfManagedServer, CCSMdwNameServer, FoNsNameServerProxy, CCSMdwRwNetProxy ***Library (Common):*** EcUr *Class:* EcUrServerUR ***Library:*** event *Class:* EcLgErrorMsg ***Process:*** EcCsRegistry *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	*Process:* EcGwDARServer *Class:* EcGwDARGatewayRequest_S	The **DCCI CSCI** provides a library of services available to each SDPS and CSMS CSCI/CSC. The CSCI services required to perform specific assignments are requested from the DCCI CSCI. These services include: • CCS Middleware Support • Database Connection Services • File Transfer Services • Network & Distributed File Services • Bulk Data Transfer Services • Name/Address Services • Password Services • Server Request Framework (SRF) • Universal Reference (UR) • Error/Event Logging • Fault Handling Services • Mode Information • Query Registry - Retrieving the requested configuration attribute-value pairs from the Configuration Registry

Table 4.8-8. ASTER DAR Gateway Server Process Interface Events (4 of 4)

Event	Event Frequency	Interface	Initiated by	Event Description
Return Configuration Parameters	One per configuration registry query	*Process:* EcGwDARServer *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	*Process:* EcCsRegistry *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	The **EcCsRegistry** returns the requested configuration parameters to the EcGwDARServer.
Push Data	One order per user request	*Process:* EcGwDARServer *Library:* IcDarApi *Class:* EcGwDARGatewayRequest_S	*Process:* EcDsStFtpServer *Library:* DsStTmClient *Class:* DsStPatron	The **EcDsStFtpServer** pushes data (i.e., Expedited Data Sets), using the FTP service and an FTP Daemon, to the EcGwDARServer for data distribution per user request. A signal file is also sent to indicate the completion of the file transfer by particular ESDTs that function this way.
Return User Profile	One per profile request	*Process:* EcGwDARServer *Class:* EcGwDARGatewayRequest_S	*Process:* EcMsAcRegUserSrvr *Libraries:* MsAcClnt, MsAcComm *Class:* EcAcProfileMgr	The EcGwDARServer receives user profile information from the **EcMsAcRegUserSrvr** to authenticate a user.

4.8.1.2.6 ASTER DAR Gateway Server Data Stores

Data stores are not applicable for the ASTER DAR Gateway.

4.8.1.3 E-mail Parser Gateway Server Software Description

4.8.1.3.1 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 an Expedited Data Sets are 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, user services personnel place Expedited Data Set subscriptions at the GSFC DAAC on behalf of the ASTER GDS. Each time the GSFC DAAC receives Expedited Data Sets

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.

4.8.1.3.2 E-mail Parser Gateway Server Context

Figure 4.8-6 is the E-mail Parser Gateway Server context diagram. Table 4.8-9 provides descriptions of the interface events shown in the E-mail Parser Gateway Server context diagram.

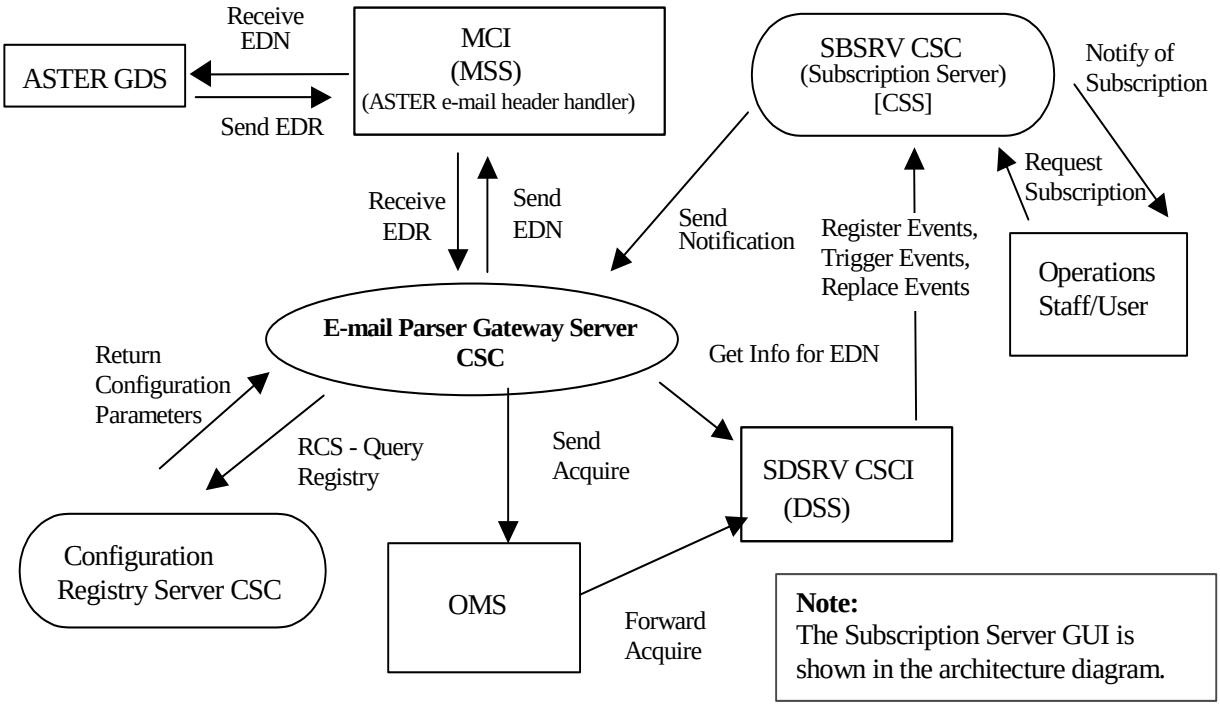


Figure 4.8-6. E-mail Parser Gateway Server Context Diagram

Table 4.8-9. E-mail Parser Gateway Server Interface Events (1 of 2)

Event	Interface Event Description
Send EDN	The E-mail Parser Gateway Server CSC stores the Expedited Data set Notification (EDN) messages with Universal References (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 Expedited Data Sets from EDOS to the ECS.

Table 4.8-9. E-mail Parser Gateway Server Interface Events (2 of 2)

Event	Interface Event Description
Register Events	The **SDSRV CSCI** sends the subscription events for an Earth Science Data Type (ESDT) 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 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.
Notify of Subscription	The SBSRV CSC sends notification (e-mail) to the **Operations Staff** when the subscribed event occurs.
Get Info for EDN	Expedited Data Set Notification 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 OMS CSCI via CCS Middleware calls. This is similar to the “Request Product” interface event, except it applies to EDOS expedited data.
Forward Acquire	OMS CI verify the EDR and forward it to SDSRV. The EDR is stored in OMS database and displayed in OMS GUI.
Request Communications Support (RCS)	The DCCI CSCI provides a library of services available to **each SDPS** and **CSMS CSCI/CSC**. The CSCI services required to perform specific assignments are requested from the DCCI CSCI. These services include: • CCS Middleware Support • Database Connection Services • File Transfer Services • Network & Distributed File Services • Bulk Data Transfer Services • Name/Address Services • Password Services • Server Request Framework (SRF) • Universal Reference (UR) • Error/Event Logging • Fault Handling Services • Mode Information • Query Registry - Retrieving the requested configuration attribute-value pairs from the Configuration Registry
Return Configuration Parameters	The **Configuration Registry CSC** returns the requested configuration parameters to the E-mail Parser Gateway Server CSC.
Receive EDN	The **MCI** E-mail header handler adds a header to the Expedited Data set Notification (EDN) and is received by the ASTER GDS via E-mail over the EOSDIS Mission Support Network (EMSn).
Send EDR	ASTER GDS personnel select the EDN needed and send an Expedited Data set Request (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.

4.8.1.3.3 E-mail Parser Gateway Server Architecture

Figure 4.8-7 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.

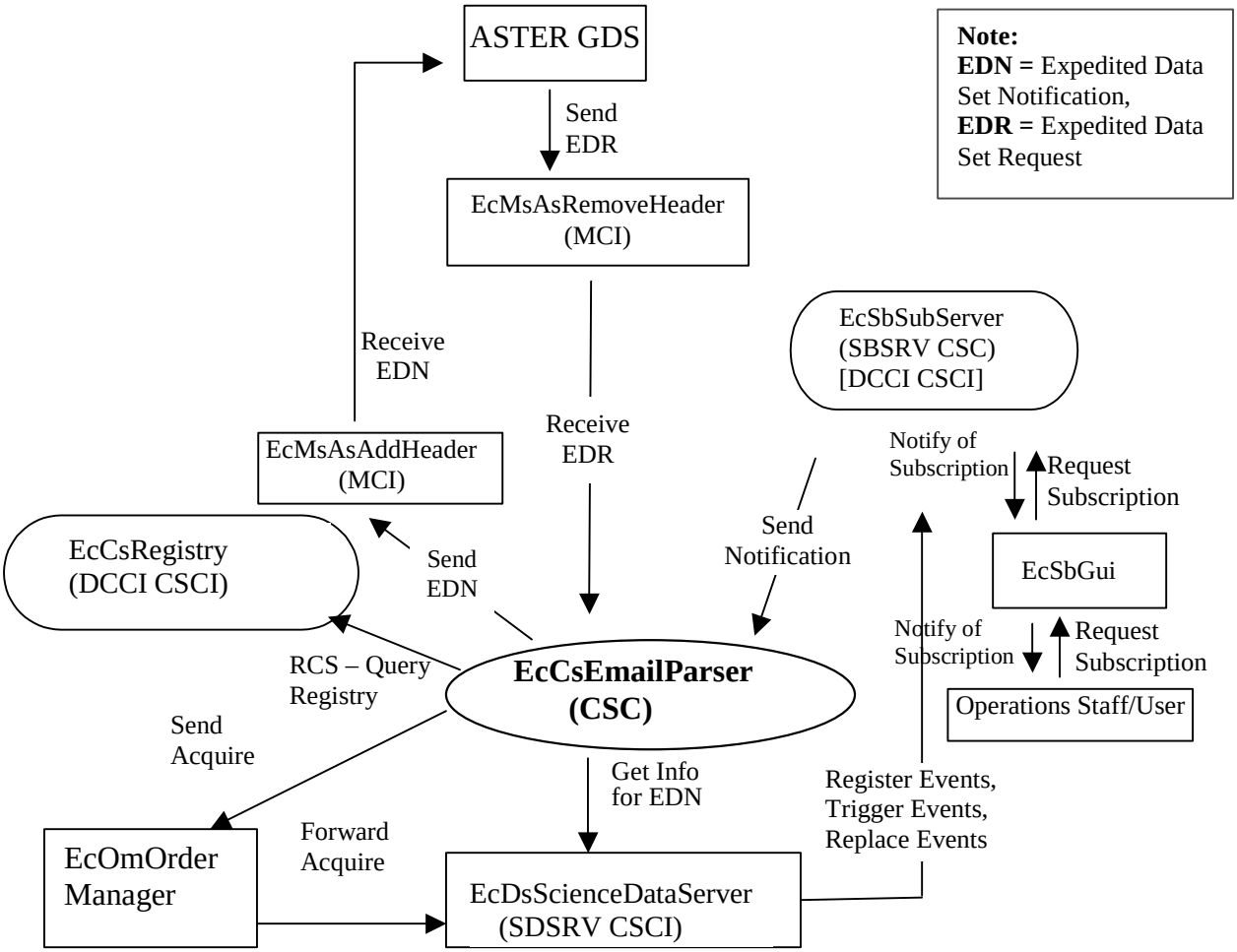


Figure 4.8-7. E-mail Parser Gateway Server Architecture Diagram

4.8.1.3.4 E-mail Parser Gateway Server Process Descriptions

Table 4.8-10 provides a description of the process shown in the E-mail Parser Gateway Server architecture diagram.

Table 4.8-10. E-mail Parser Gateway Server Processes

Process	Type	COTS/Developed	Functionality
EcCsEmailParser	Server	Developed	• Get Universal Reference (UR) from the Subscription Notification and use this UR to get the information for Expedited Data set Notification (EDN) from the Science Data Server and send it to the Add Header script for notifying ASTER GDS. • Get Expedited Data set Request (EDR) from the Remove Header script. • Store and parse subscriptions and EDR in /EDN, /EDR directory • Send an Acquire request to the Order Manager Server via a CCS Middleware call for an EDR request.

4.8.1.3.5 E-mail Parser Gateway Server Process Interface Descriptions

Table 4.8-11 provides descriptions of the interface events shown in the E-mail Parser Gateway Server architecture diagram.

Table 4.8-11. E-mail Parser Gateway Server Process Interface Events (1 of 4)

Event	Event Frequency	Interface	Initiated by	Event Description
Send EDR	One per EDR send	*Script:* EcMsAsRemoveHeader	ASTER GDS Operations Staff *Process:* EcCsEmailParser *Library:* EcCsEmailParser	After selecting the Expedited Data set Notification (EDN), the ASTER GDS personnel send an Expedited Data set Request (EDR) to the **EcMsAsRemoveHeader** script to have the header removed.
Receive EDR	One per e-mail send from ASTER GDS	*Script:* 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	*Process:* EcCsEmailParser *Class:* EcCsEmailParser	*Process:* EcSbSubServer *Library:* CsEmMailRelA *Class:* CsEmMailRelA	The **EcSbSubServer** sends an EDN via E-mail to the EcCsEmailParser.

Table 4.8-11. E-mail Parser Gateway Server Process Interface Events (2 of 4)

Event	Event Frequency	Interface	Initiated by	Event Description
Request Subscription	One per subscription request	*Process:* EcSbSubServer *Libraries:* EcSbSr, EcSbCl *Classes:* EcSbSubmitSubRequest, EcSbSubscription, EcClSubscription	Operations Staff/User *Process:* EcSbGui *Class:* 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.
Notify of Subscription	One per subscription submitted	*Process:* EcSbGui *Class:* EcSbSubscriptionDispatcher	*Process:* EcSbSubServer *Library:* EcSbSr *Classes:* EcSbSubscription, EcSbNotification	The EcSbSubServer sends e-mail to the **Operations Staff** (via the **EcSbGui**).
Register Events	One per ESDT installation	*Process:* EcSbSubServer *Library:* EcSbSrSh *Class:* EcSbEvent	*Process:* EcDsScienceDataServer *Class:* DsDeEventCustomizer	The **EcDsScienceDataServer** sends the subscription events for an Earth Science Data Type (ESDT) 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	*Process:* EcSbSubServer *Library:* EcSbCl *Class:* EcClEvent	*Process:* EcDsScienceDataServer *Class*: 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	*Process:* EcSbSubServer *Library:* EcSbSrSh *Class:* EcClEvent	*Process:* EcDsScienceDataServer *Class:* DsDeEventCustomizer	The **EcDsScienceDataServer** sends the updated subscription events for an ESDT to the EcSbSubServer when an ESDT is updated. This event replaces the original event in the EcSbSubServer.

Table 4.8-11. E-mail Parser Gateway Server Process Interface Events (3 of 4)

Event	Event Frequency	Interface	Initiated by	Event Description
Get Info for EDN	One per notification of the ASTER GDS	*Process:* EcDsScienceDataServer *Library:* DsCl *Class:* DsClESDTReference	*Process:* EcCsEmailParser *Class:* 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.
Send Acquire	One per acquire created	*Process:* *EcOmOrderManager* *or* EcDsScienceDataServer *Library:* DsCl *Class:* DsClRequest	*Process:* EcCsEmailParser *Class:* EcCsEmailParser	An “acquire” (instruction to obtain data) is created by the EcCsEmailParser and sent to EcOmOrderManager when configured to submit EDR to OMS or sent to **EcDsScienceDataServer** when configured to submit EDR to SDSRV via CCS Middleware calls. This is similar to the “Request Product” interface event, except it applies to EDOS expedited data.

Table 4.8-11. E-mail Parser Gateway Server Process Interface Events (4 of 4)

Event	Event Frequency	Interface	Initiated by	Event Description
Request Communications Support	One service per request	***Process:*** EcCsIdNameServer ***Libraries:*** EcPf, Middleware, FoNs, FoIp, oodce *Classes:* EcPfManagedServer, CCSMdwNameServer, FoNsNameServerProxy, CCSMdwRwNetProxy ***Library (Common):*** EcUr *Class:* EcUrServerUR ***Library:*** event *Class:* EcLgErrorMsg ***Process:*** EcCsRegistry *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	*Process:* EcCsEmailParser *Library:* EcCsEmailParser	The DCCI CSCI provides a library of services available to each SDPS and CSMS CSCI/CSC. The CSCI services required to perform specific assignments are requested from the DCCI CSCI. These services include: • CCS Middleware Support • Database Connection Services • File Transfer Services • Network & Distributed File Services • Bulk Data Transfer Services • Name/Address Services • Password Services • Server Request Framework (SRF) • Universal Reference (UR) • Error/Event Logging • Fault Handling Services • Mode Information • Query Registry - Retrieving the requested configuration attribute-value pairs from the Configuration Registry
Receive EDN	One per e-mail send from ECS	ASTER GDS	*Script:* 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	*Script:* EcMsAsAddHeader	*Process:* EcCsEmailParser *Class:* EcCsEmailParser	The EcCsEmailParser sends the Send EDN to the **EcMsAsAddHeader** script to have a header added.

4.8.1.3.6 E-mail Parser Gateway Server Data Stores

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

4.8.1.4 MOJO Gateway Server Computer Software Component Software Description

4.8.1.4.1 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 three major groups:

1. 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 subscriptions, submission/modification of DARs, DAR queries, and requests for user profiles via proxy objects
2. The MOJO Gateway Server provides a security gateway to various ECS services within MSS, which can:
   - Provide an abstract login interface to clients to initiate a new session
   - Spawn proxy objects to process requests from clients
   - Do CCS Middleware calls to access ECS services on behalf of clients
3. 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

4.8.1.4.2 MOJO Gateway Server Context

Figure 4.8-8 is the MOJO Gateway Server context diagrams. Table 4.8-12 provides descriptions of the interface events in the MOJO Gateway Server context diagrams.

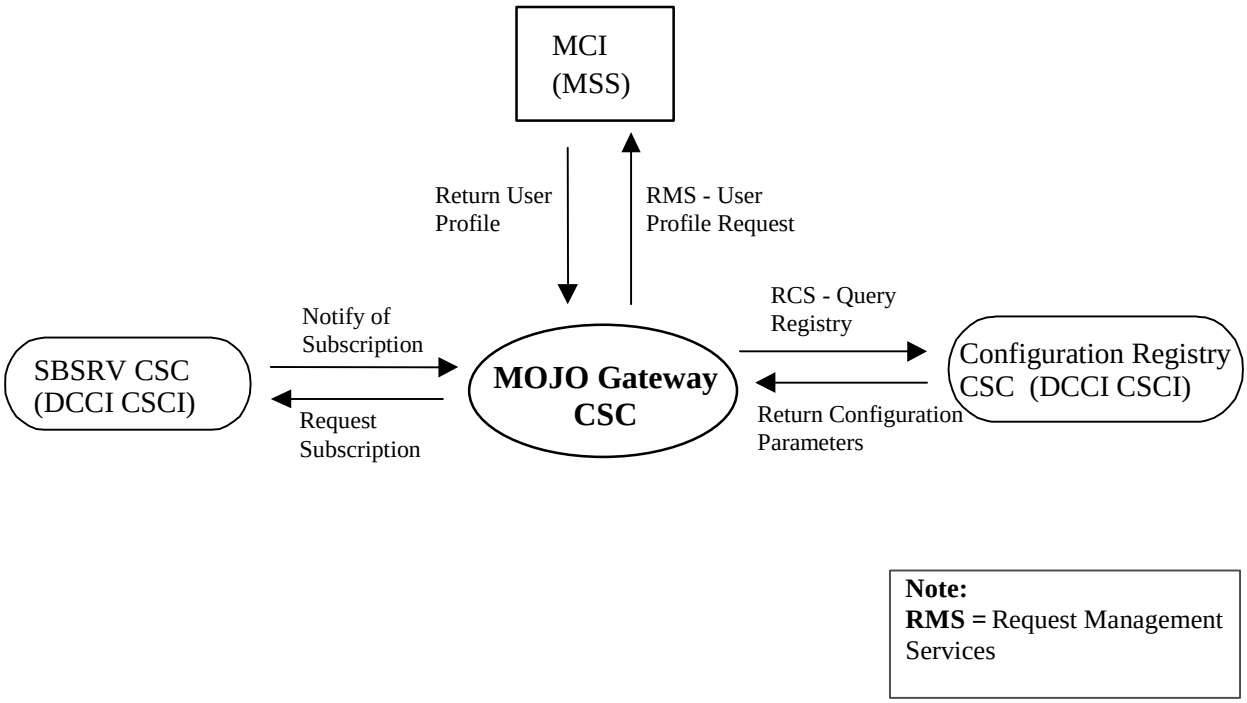


Figure 4.8-8. MOJO Gateway Server Context Diagram

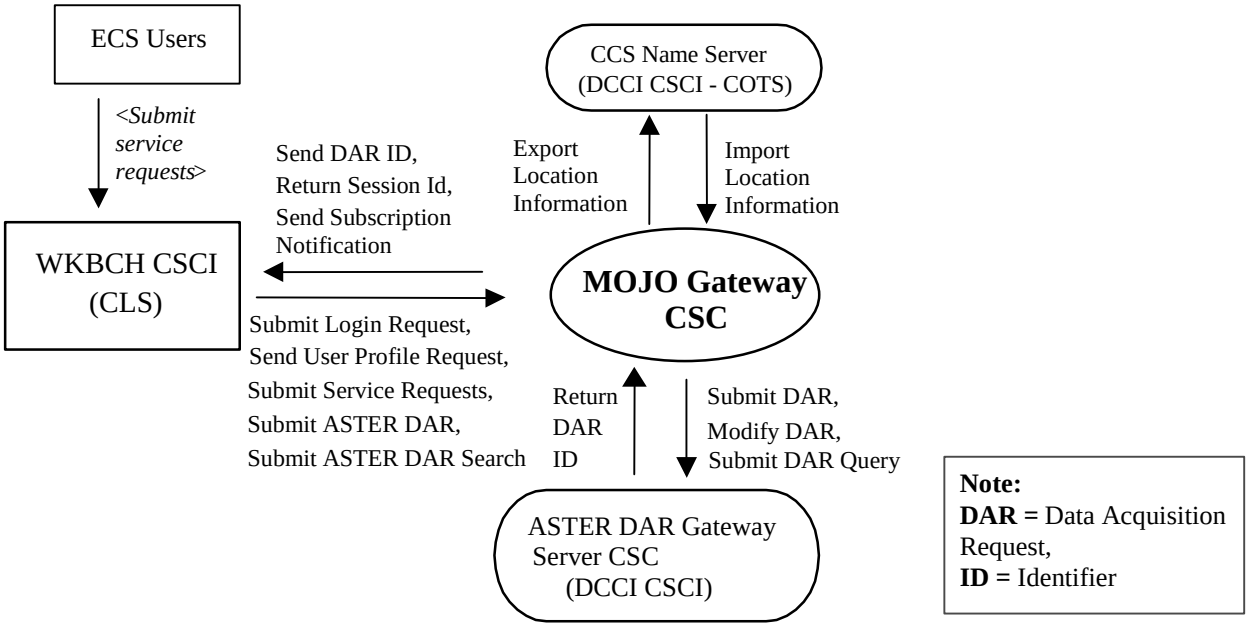


Figure 4.8-8. MOJO Gateway Server Context Diagram (cont.)

Table 4.8-12. MOJO Gateway Server Interface Events (1 of 2)

Event	Interface Event Description
Request Management Services	The **MCI** provides 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: • **System startup and shutdown -** Please refer to the release-related, current version of the Mission Operations Procedures for the ECS Project document (611) and the current ECS Project Training Material document (625), identified in Section 2.2.1 of this document. • **User Profile Request** - The MCI provides requesting subsystems with User Profile information such as e-mail address and shipping address upon request by authorized users to support their processing activities.
Request Communications Support (RCS)	The DCCI CSCI provides a library of services available to **each SDPS** and **CSMS CSCI/CSC**. The CSCI services required to perform specific assignments are requested from the DCCI CSCI. These services include: • CCS Middleware Support • Database Connection Services • File Transfer Services • Network & Distributed File Services • Bulk Data Transfer Services • Name/Address Services • Password Services • Server Request Framework (SRF) • Universal Reference (UR) • Error/Event Logging • Fault Handling Services • Mode Information • Query Registry - Retrieving the requested configuration attribute-value pairs from the Configuration Registry
Return Configuration Parameters	The **Configuration Registry CSC** returns the requested configuration parameters to the MOJO Gateway Server CSC.
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.
Notify of Subscription .	The **SBSRV CSC** sends notification to the MOJO Gateway Server CSC when the subscribed event occurs.
Return User Profile	The MOJO Gateway Server CSC receives a user profile from the **MCI**.
Export Location Information	The MOJO Gateway CSC sends physical and logical server location information to the CCS Name Server.
Import Location Information	The MOJO Gateway CSC receives physical and logical server location information from the CCS Name Server.
Submit DAR .	The MOJO Gateway Server CSC submits a Data Acquisition Request (DAR) to the **ASTER DAR Gateway Server 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.

Table 4.8-12. MOJO Gateway Server Interface Events (2 of 2)

Event	Interface Event Description
Modify DAR .	The MOJO Gateway Server CSC submits a modification to an existing DAR to the **ASTER DAR Gateway Server 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.
Submit DAR Query .	Submit a DAR query to the **ASTER DAR Gateway Server CSC**. The query can be of type queryxARContents, queryxARSummary, or queryxARScenes. The ASTER DAR Gateway returns the query results.
Return DAR ID .	The **ASTER DAR Gateway Server CSC** extracts the returned DAR ID and sends it to the Java DAR Tool, via the MOJO Gateway Server.
Submit Login Request	The MOJO Gateway CSC receives the user name and password from the **WKBCH CSCI** for use of ECS data and services.
Send User Profile Request	The MOJO Gateway Server CSC receives a request for a user profile from the **WKBCH CSCI**.
Submit Service Requests	The **WKBCH CSCI** submits a request for an ECS service on behalf of the user to the MOJO Gateway CSC. Service types include Subscriptions, User Profile retrievals, and DAR submittal or modification.
Submit ASTER DAR .	A user submits a request to the MOJO Gateway CSC to have ASTER data taken (a data acquisition request or DAR) using the parameters entered into the Java DAR Tool (**WKBCH CSCI**). DAR parameters are required for submittal of DARs as specified in the ASTER GDS IRD/ICD.
Submit ASTER DAR Search	The **WKBCH CSCI** (Java DAR Tool) sends, to the MOJO Gateway CSC, the request to search the ASTER DAR database by DAR parameters or a specific DAR ID for a scene of interest (to the user) from the ASTER instrument.
Send DAR ID .	As the result of a successfully submitted DAR by the MOJO Gateway CSC, the **WKBCH CSCI** receives a DAR ID. The DAR ID is a string of characters used to track a DAR.
Return Session Id .	The **WKBCH CSCI** receives a session id from the MOJO Gateway CSC to communicate with the ASTER GDS.
Send Subscription Notification	The **WBCH CSCI** receives notification via the MOJO Gateway Server CSC (from the SBSRV CSC) when the subscribed event occurs.

4.8.1.4.3 MOJO Gateway Server Architecture

Figure 4.8-9 is the MOJO Gateway Server architecture diagrams. The diagrams show the events sent to the MOJO Gateway Server process and the events the MOJO Gateway Server process sends to other processes.

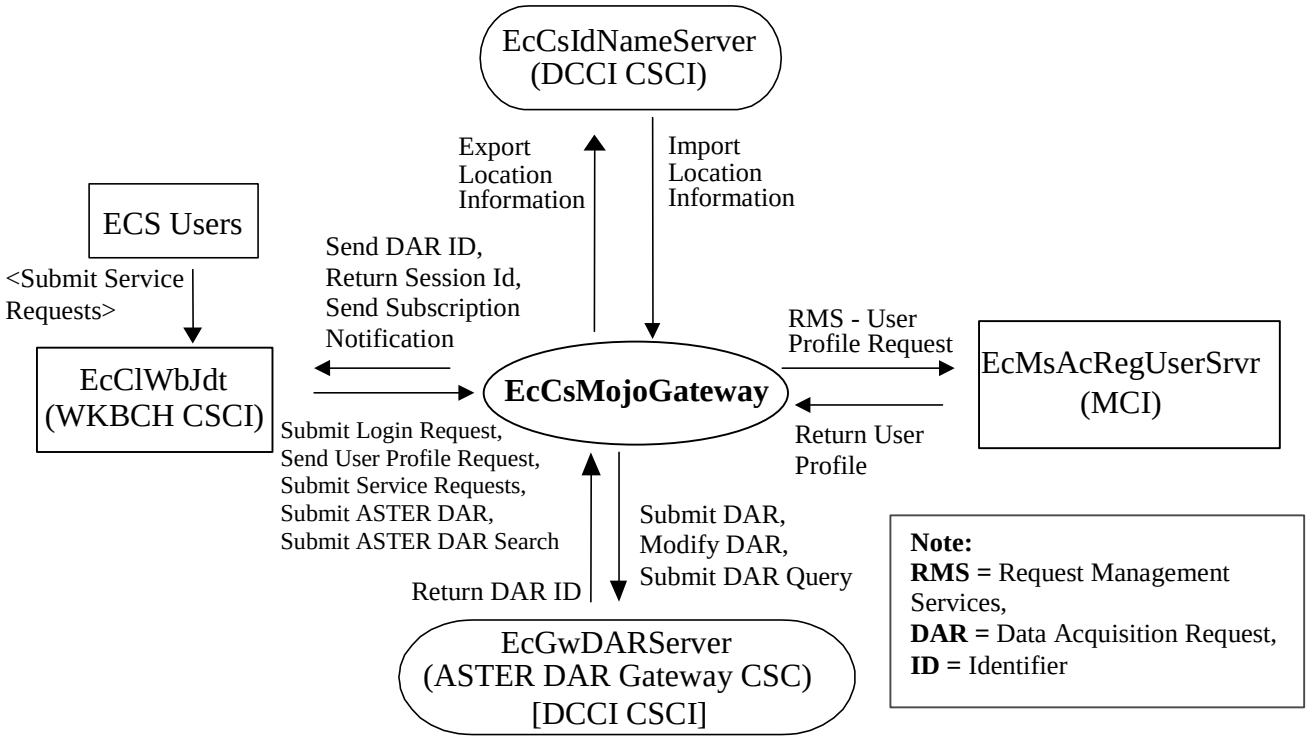


Figure 4.8-9. MOJO Gateway Server Architecture Diagram

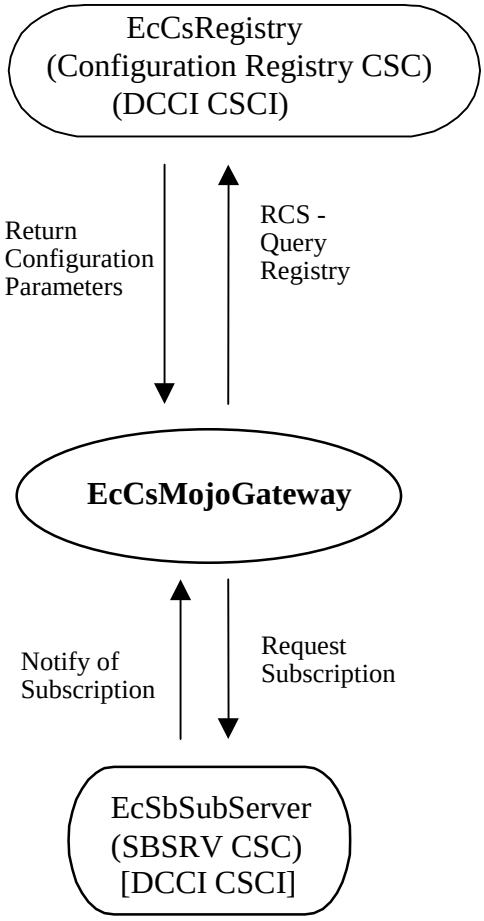


Figure 4.8-9. MOJO Gateway Server Architecture Diagram (cont.)

4.8.1.4.4 MOJO Gateway Server Process Descriptions

Table 4.8-13 provides a description of the processes in the MOJO Gateway Server architecture diagram.

Table 4.8-13. MOJO Gateway Server Processes

Process	Type	COTS / Developed	Functionality
EcCsMojoGateway	Server	Developed	The EcCsMojoGateway is a server that generally provides an internet gateway from JAVA WEB users to CCS Middleware-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 abstract interface for 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. As a server developed by ECS, it provides the following major interface functionality: • MjGetJava WebMessage: Reads a message from JESS • MjProcessRequest: This is a polymorphic method, which defines a common interface and by which the various ECS service facilities are invoked • MjSendJava WebMessage: Sends a message to JESS The EcCsMojoGateway supports: • Multiple concurrent requests

4.8.1.4.5 MOJO Gateway Server Process Interface Descriptions

Table 4.8-14 provides descriptions of the interface events shown in the MOJO Gateway Server architecture diagrams.

Table 4.8-14. MOJO Gateway Server Process Interface Events (1 of 6)

Event	Event Frequency	Interface	Initiated by	Event Description
Import Location Information	As required for processing	*Process:* EcCsMojoGateway *Library:* EcCsMojoGateway *Class:* EcMjManagedSrv	*Process:* EcCsIdNameServer *Libraries:* EcPf, Middleware, FoNs, FoIp, oodce *Classes:* EcPfManagedServer, CCSMdwNameServer, FoNsNameServerProxy, CCSMdwRwNetProxy	The **EcCsIdNameServer** sends server location information to ECS applications by request.
Request Management Services (RMS)	One per notice received	*Process:* EcMsAcRegUserSrvr *Library:* MsAcClnt *Classes:* MsAcUserProfile, RWPortal	*Process:* EcCsMojoGateway *Class*: EcMjRetrieveProfileProxy	**User Profile Request** – The **EcMsAcRegUserSrvr** provides requesting processes with User Profile parameters such as e-mail and shipping addresses to support their processing activities.
Return User Profile	Per user request	*Process:* EcCsMojoGateway *Library:* EcCsMojoGateway *Class:* EcMjRetrieveProfileProxy	*Process:* EcMsAcRegUserSrvr *Libraries:* MsAcClnt, MsAcComm *Class:* EcAcProfileMgr	The EcCsMojoGateway receives user profile information from the EcMsAcRegUserSrvr.
Submit DAR	Per user request	*Process:* EcGwDARServer *Library:* EcGwDAR *Class:* EcGwDARSubmitDarRequest_C	*Process:* EcCsMojoGateway *Library:* EcCsMojoGateway *Class:* 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 4.8-14. MOJO Gateway Server Process Interface Events (2 of 6)

Event	Event Frequency	Interface	Initiated by	Event Description
Modify DAR	Per user request	*Process:* EcGwDARServer *Library:* EcGwDAR *Class:* EcGwDARModifyDarRequest_C	*Process:* EcCsMojoGateway *Library:* EcCsMojoGateway *Class:* 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.
Submit DAR Query	Per user request	*Process:* EcGwDARServer *Library:* EcGwDAR *Classes:* EcGwDARQueryxARContentsRequest_C, EcGwDARQueryxARSummaryRequest_C, EcGwDARQueryxARScenesRequest_C	*Process:* EcCsMojoGateway *Library:* EcCsMojoGateway *Classes:* EcMjDarQueryxARContentsProxy, EcMjDarQueryxARSummaryProxy, EcMjDarQueryxARScenesProxy	The EcCsMojoGateway sends a query request to the EcGwDARServer. The EcGwDARServer returns the query results.
Return DAR ID	One per DAR ID sent	*Process:* EcCsMojoGateway *Library:* EcGwDAR *Class:* EcGwDARSubmitDarRequest_C	*Process:* EcGwDARServer *Class:* EcGwDARGatewayRequest_S	The EcGwDARServer extracts the returned DAR ID and sends it to the Java DAR Tool, via the EcCsMojoGateway.
Submit Login Request	One per User	*Process:* EcCsMojoGateway *Class:* EcMjDCELogin Proxy	*Process:* EcClWbJdt *Class:* ClWbUrUserInfo	The **EcClWbJdt** sends the user name and password to the EcCsMojoGateway for use of ECS data and services.
Send User Profile Request	One per request	*Process:* EcCsMojoGateway *Class:* EcMjRetrieveProfileProxy	*Process:* EcClWbJdt *Class:* JDTApplet	The **EcClWbJdt** sends user profile requests to the EcCsMojoGateway to get user profile information for DAR submit authorization based upon the user id information provided by the user.

Table 4.8-14. MOJO Gateway Server Process Interface Events (3 of 6)

Event	Event Frequency	Interface	Initiated by	Event Description
Submit Service Requests	One type per user request	*Process:* EcCsMojoGateway *Library:* EcCsMojoGateway *Class:* EcMjECSSbsrvProxy	*Process:* EcClWbJdt *Library:* Standard JDK 1.1.x socket support *Class:* Java.net.Socket	The **EcClWbJdt** submits a request for an ECS service on behalf of users to the EcCsMojoGateway. The only service type available is the Subscription request.
Submit ASTER DAR	One per request	*Process:* EcCsMojoGateway *Classes:* EcMjDarSubmitDarProxy, EcMjDarModifyDarProxy	*Process:* EcClWbJdt *Class:* JDTApplet	A user submits a request to the EcCsMojoGateway to have ASTER data taken (a data acquisition request or DAR) using the parameters entered into the Java DAR Tool (**EcClWbJdt**). DAR parameters are required for submittal of DARs as specified in the ASTER GDS IRD/ICD.
Submit ASTER DAR Search	One per set of DAR parameters or DAR ID	*Process:* EcCsMojoGateway Class: EcMjDarQueryxARScenesProxy	*Process:* EcClWbJdt Class: JDTApplet	The **EcClWbJdt** process sends, to the EcCsMojoGateway, the request to search the ASTER DAR database by DAR parameters or a specific DAR ID for a scene of interest (to the user) from the ASTER instrument.
Send DAR ID	One per set of DAR parameters	*Process:* EcClWbJdt *Class:* JDTApplet	*Process:* EcCsMojoGateway *Library:* EcCsMojoGateway *Class:* EcMjDarQueryxARScenesProxy	As the result of a successfully submitted DAR, the **EcClWbJdt** receives a DAR ID from the EcCsMojoGateway. This is a string of characters used to track a DAR. The user receives notification every time data resulting from this DAR is received by the ECS.

Table 4.8-14. MOJO Gateway Server Process Interface Events (4 of 6)

Event	Event Frequency	Interface	Initiated by	Event Description
Return Session Id	One per authentication request	*Process:* EcClWbJdt *Class:* JDTApplet	*Process:* EcCsMojoGateway *Class:* EcMjDCELoginProxy	The EcCsMojoGateway returns a session id to the **EcClWbJdt** for the user to communicate between the EcClWbJdt (DAR Tool) and the EcCsMojoGateway.
Send Subscription Notification	One per subscription request	*Process:* EcClWbJdt *Class:* JDTApplet	*Process:* EcCsMojoGateway *Library:* EcCsMojoGateway *Class:* EcMjDarQueryxARScenesProxy	The EcCsMojoGateway sends the subscription notification to the **EcClWbJdt** (Java DAR Tool) to notify a user of a subscription being satisfied.
Export Location Information	Once at system startup and after each failure recovery	*Process:* EcCsIdNameServer *Libraries:* EcPf, Middleware, FoNs, FoIp, oodce *Classes:* EcPfManagedServer, CCSMdwNameServer, FoNsNameServerProxy, CCSMdwRwNetProxy	*Process:* EcCsMojoGateway *Library:* EcCsMojoGateway *Class:* EcMjManagedSrv	The EcCsMojoGateway sends physical and logical server location information to the **EcCsIdNameServer**.

Table 4.8-14. MOJO Gateway Server Process Interface Events (5 of 6)

Event	Event Frequency	Interface	Initiated by	Event Description
Request Communications Support	One service per request	***Process:*** EcCsIdNameServer ***Libraries:*** EcPf, Middleware, FoNs, FoIp, oodce *Classes:* EcPfManagedServer, CCSMdwNameServer, FoNsNameServerProxy, CCSMdwRwNetProxy **Library (Common):** EcUr *Class:* EcUrServerUR ***Library:*** event *Class:* EcLgErrorMsg ***Process:*** EcSbSubServer *Library:* EcSbCl Classes: EcClEvent, EcClTriggerEventCb, EcClRegisterEventCb ***Process:*** EcCsRegistry *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	*Process:* EcCsMojoGateway *Library:* EcCsMojoGateway *Class:* EcMjManagedSrv	The DCCI CSCI provides a library of services available to each SDPS and CSMS process. The process services required to perform specific assignments are requested from the DCCI CSCI. These services include: • CCS Middleware Support • Database Connection Services • File Transfer Services • Network & Distributed File Services • Bulk Data Transfer Services • Name/Address Services • Password Services • Server Request Framework (SRF) • Universal Reference (UR) • Error/Event Logging • Fault Handling Services • Mode Information • Query Registry - Retrieving the requested configuration attribute-value pairs from the Configuration Registry

Table 4.8-14. MOJO Gateway Server Process Interface Events (6 of 6)

Event	Event Frequency	Interface	Initiated by	Event Description
Request Subscription	Per user request	*Process:* EcSbSubServer *Library:* EcSbCl *Class:* EcClSubscription	*Process:* EcCsMojoGateway *Library:* EcCsMojoGateway *Class:* 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.
Notify of Subscription	Per user request	*Process:* EcCsMojoGateway *Library:* EcCsMojoGateway *Class:* EcMjECSSbsrvProxy	*Process:* EcSbSubServer *Library:* EcSbCl *Class:* EcClSubscription	The EcSbSubServer subscription identifier is not returned to the user, but used by the EcCsMojoGateway to determine what subscription event has been satisfied. The subscription notification is passed from the EcCsMojoGateway to the EcClWbJdt for the user.
Return Configuration Parameters	One per configuration registry query	*Process:* EcCsMojoGateway *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	*Process:* EcCsRegistry	The EcCsRegistry returns the requested configuration parameters to the EcCsMojoGateway.

4.8.1.4.6 MOJO Gateway Server Data Stores

Data stores are not applicable for the MOJO Gateway Server.

4.8.1.5 Configuration Registry Server Software Description

4.8.1.5.1 Configuration Registry Server Functional Overview

The Configuration Registry Server provides an interface to retrieve configuration attribute-value pairs and another interface to retrieve distribution options for ECS Servers from the Configuration Registry Database, via a Sybase ASE. The Configuration Registry Server maintains an internal representation of the tree in which configuration attribute-value pairs and distribution options 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 not only accepts queries to the Configuration Registry Database with a configuration path and returns a list of attribute-value pairs, but also accepts queries of distribution options to the Configuration Registry Database with an ESDT short name and version and returns a hierarchical list of attributes. 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.

4.8.1.5.2 Configuration Registry Server Context

Figure 4.8-10 is the Configuration Registry Server context diagrams. Table 4.8-15 provides descriptions of the interface events in the Configuration Registry Server context diagrams.

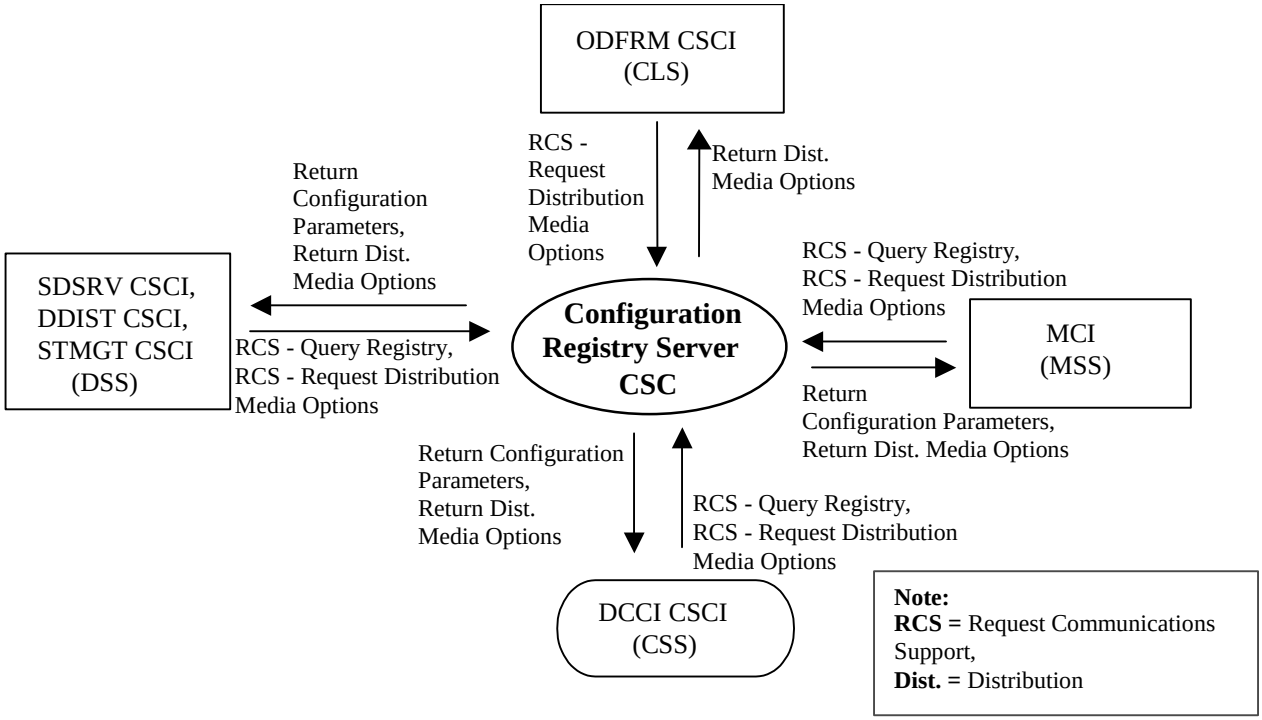


Figure 4.8-10. Configuration Registry Server Context Diagram

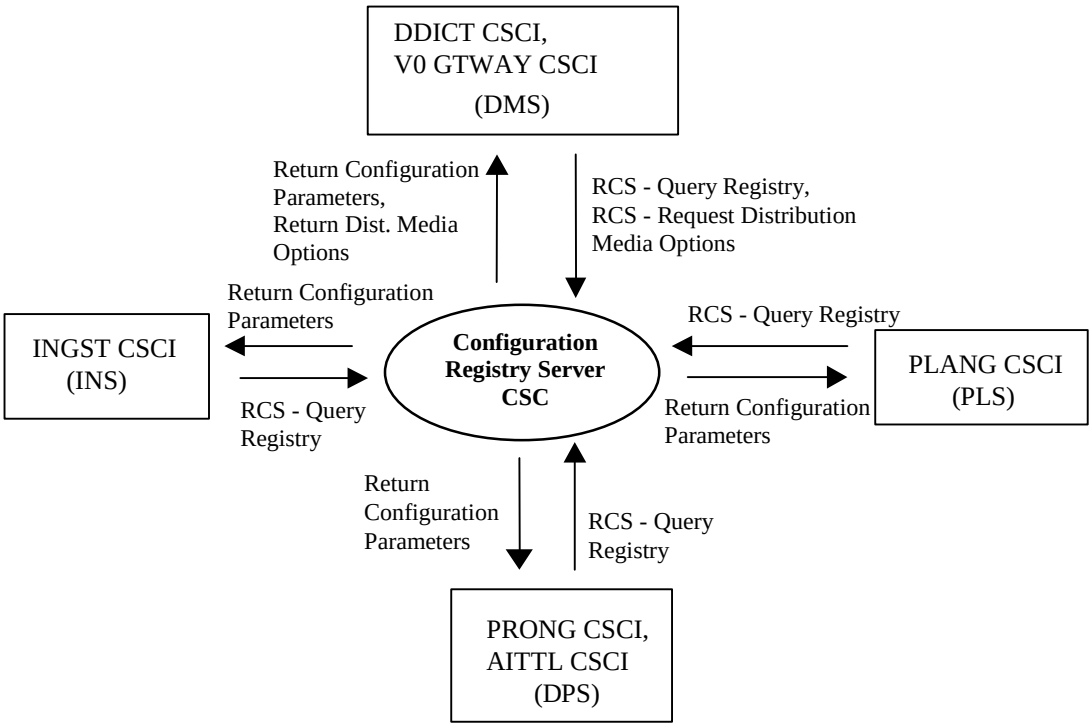


Figure 4.8-10. Configuration Registry Server Context Diagram (cont.)

Table 4.8-15. Configuration Registry Server Interface Events (1 of 2)

Event	Interface Event Description
RCS - Request Distribution Media Options	The **ODFRM**, **MCI**, **SDSRV**, **DDIST**, **STMGT** and other **DCCI CSCI CSCs** query the Configuration Registry Server for configuration parameters. The ECS Servers pass in an ESDT short name and version. The Registry Server uses this information as a starting point in the tree and returns all distribution options associated with it.
RCS - Query Registry	Upon startup, the **MCI**, **SDSRV**, **DDIST**, **STMGT** and other **DCCI CSCI CSCs** 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.
Return Configuration Parameters	The Configuration Registry CSC returns the requested configuration parameters to the **SDSRV, DDIST, STMGT, MCI and DCCI,** CSCIs.
Return Dist. Media Options	The Configuration Registry CSC returns the requested distribution media options to the ODFRM, **SDSRV, DDIST, STMGT, MCI and DCCI** CSCIs.

Table 4.8-15. Configuration Registry Server Interface Events (2 of 2)

Event	Interface Event Description
RCS - Request Distribution Media Options	The **DDICT and V0 GTWAY CSCIs** query the Configuration Registry Server for configuration parameters. The ECS Servers pass in an ESDT short name and version. The Registry Server uses this information as a starting point in the tree and returns all distribution options associated with it.
RCS - Query Registry	Upon startup, the **DDICT, V0 GTWAY, PLANG, PRONG, AITTL and INGST CSCIs** 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.
Return Configuration Parameters	The Configuration Registry CSC returns the requested configuration parameters to the **DDICT, V0 GTWAY, PLANG, PRONG, AITTL, and INGST** CSCIs.
Return Dist. Media Options	The Configuration Registry CSC returns the requested distribution media options to the **DDICT and V0 GTWAY** CSCIs.

4.8.1.5.3 Configuration Registry Server Architecture

Figure 4.8-11 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.

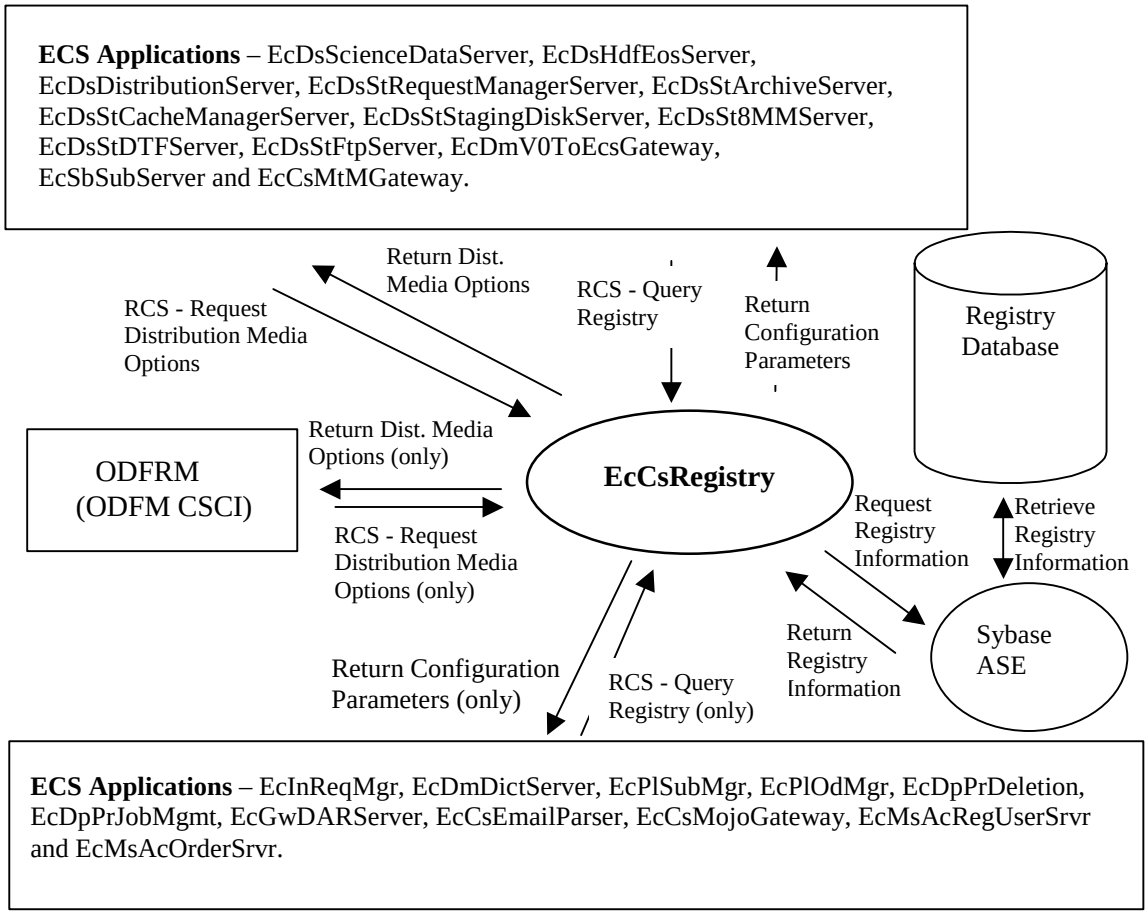


Figure 4.8-11. Configuration Registry Server Architecture Diagram

4.8.1.5.4 Configuration Registry Server Process Descriptions

Table 4.8-16 provides a description of the processes in the Configuration Registry Server architecture diagram.

Table 4.8-16. Configuration Registry Server Processes

Process	Type	COTS / Developed	Functionality
EcCsRegistry	Server	Developed	The Configuration Registry Server provides an interface to retrieve configuration attribute-value pairs (the returned attribute-value pairs are stored in cache memory on the PF client side). The Configuration Registry Server provides an interface to retrieve distribution options for ECS Servers from the Configuration Registry Database, via a Sybase ASE. The Configuration Registry Server not only accepts queries to the Configuration Registry Database with a configuration path and returns a list of attribute-value pairs, but also accepts queries of distribution options to the Configuration Registry Database with an ESDT short name and version and returns a list of attributes. A wild-card character may be specified as the last element in the path to retrieve all attributes in the sub-tree specified. The Configuration Registry Server provides another interface to retrieve external data subsetters for Synergy.

4.8.1.5.5 Configuration Registry Server Process Interface Descriptions

Table 4.8-17 provides descriptions of the interface events shown in the Configuration Registry Server architecture diagram.

Table 4.8-17. Configuration Registry Server Process Interface Events (1 of 4)

Event	Event Frequency	Interface	Initiated by	Event Description
RCS - Query Registry	One per client request	*Process:* EcCsRegistry *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	*Processes:* EcDsScienceDataServer, EcDsHdfEosServer, EcDsDistributionServer, EcDsStRequestManagerServer, EcDsStArchiveServer, EcDsStCacheManagerServer, EcDsStStagingDiskServer, EcDsSt8MMServer, EcDsStDTFServer, EcDsStFtpServer, EcDmV0ToEcsGateway, EcSbSubServer, EcGwDARServer, EcCsMtMGateway	An **ECS application** (process) sends a query request to the Configuration Server to retrieve a list of attribute-value pairs (configuration parameters) needed by the application.

Table 4.8-17. Configuration Registry Server Process Interface Events (2 of 4)

Event	Event Frequency	Interface	Initiated by	Event Description
Return Configuration Parameters	One set per request	*Processes:* EcDsScienceDataServer, EcDsHdfEosServer, EcDsDistributionServer, EcDsStRequestManagerServer, EcDsStArchiveServer, EcDsStCacheManagerServer, EcDsStStagingDiskServer, EcDsSt8MMServer, EcDsStDTFServer, EcDsStFtpServer, EcDmV0ToEcsGateway, EcSbSubServer, EcGwDARServer, EcCsMtMGateway	*Process:* EcCsRegistry *Library:* EcCsRegistry Class: EcRgRegistryServer_C	The EcCsRegistry returns the attribute-value pairs (configuration parameters) to the various **ECS applications** (processes) upon request.
Retrieve Registry Information	Per client request	Registry Database	Sybase ASE (COTS)	The **Sybase ASE** receives the request and retrieves the necessary attribute-value pairs and returns them to the EcCsRegistry.
Request Registry Information	Per client request	Sybase ASE (COTS)	*Process:* EcCsRegistry *Class:* EcRgRegistryServer_S	The Configuration Server sends the request to the **Sybase ASE** to retrieve the attribute-value pairs.
Return Registry Information	Per client request	*Process:* EcCsRegistry Class: EcRgRegistryServer_S	Sybase ASE (COTS)	The Configuration Server receives the registry information (attribute-value pairs) from the **Sybase ASE**.

Table 4.8-17. Configuration Registry Server Process Interface Events (3 of 4)

Event	Event Frequency	Interface	Initiated by	Event Description
RCS - Query Registry (only)	One per client request	*Process:* EcCsRegistry *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	*Processes:* EcInReqMgr, EcDmDictServer, EcPlSubMgr, EcPlOdMgr, EcDpPrDeletion, EcDpPrJobMgmt, EcGwDARServer, EcCsEmailParser, EcCsMojoGateway, EcMsAcRegUserSrvr, EcMsAcOrderSrvr	An **ECS application** (process) sends a query request to the Configuration Server to retrieve a list of attribute-value pairs (configuration parameters) needed by the application.
Return Configuration Parameters (only)	One set per request	*Processes:* EcInReqMgr, EcDmDictServer, EcPlSubMgr, EcPlOdMgr, EcDpPrDeletion, EcDpPrJobMgmt, EcGwDARServer, EcCsEmailParser, EcCsMojoGateway, EcMsAcRegUserSrvr, EcMsAcOrderSrvr	*Process:* EcCsRegistry *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	The EcCsRegistry returns the attribute-value pairs (configuration parameters) to the various **ECS applications** (processes) upon request.
RCS - Request Distribution Media Options (only)	One per client request	*Process:* EcCsRegistry *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	*Process:* ODFRM (CGI Interface)	The **ODFRM,** via the CGI Interface, requests the distribution media options from the EcCsRegistry.
Return Distribution Media Options (only)	One per client request	*Process:* ODFRM (CGI Interface)	*Process:* EcCsRegistry *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	The **ODFRM,** via the CGI Interface, receives the distribution media options from the EcCsRegistry.

Table 4.8-17. Configuration Registry Server Process Interface Events (4 of 4)

Event	Event Frequency	Interface	Initiated by	Event Description
RCS - Request Distribution Media Options	One per client request	*Process:* EcCsRegistry *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	*Processes:* EcDsScienceDataServer, EcDsHdfEosServer, EcDsDistributionServer, EcDsStRequestManagerServer, EcDsStArchiveServer, EcDsStCacheManagerServer, EcDsStStagingDiskServer, EcDsSt8MMServer, EcDsStDTFServer, EcDsStFtpServer, EcDmV0ToEcsGateway, EcSbSubServer, EcCsMtMGateway	An **ECS application** (process) sends a query of distribution options to the Configuration Server to retrieve a list of distribution options.
Return Dist. Media Options	One set per request	*Processes:* EcDsScienceDataServer, EcDsHdfEosServer, EcDsDistributionServer, EcDsStRequestManagerServer, EcDsStArchiveServer, EcDsStCacheManagerServer, EcDsStStagingDiskServer, EcDsSt8MMServer, EcDsStDTFServer, EcDsStFtpServer, EcDmV0ToEcsGateway, EcSbSubServer, EcCsMtMGateway	*Process:* EcCsRegistry *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	The EcCsRegistry returns the distribution media options (tape) to the various **ECS applications** (processes) upon request.

4.8.1.5.6 Configuration Registry Server Data Stores

The Configuration Registry Server uses a Sybase ASE 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 4.8-18 provides descriptions of the data found in the separate Sybase ASE 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, 311-CD-628).

Table 4.8-18. Configuration Registry Server Data Stores (1 of 2)

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 (organization), 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.

Table 4.8-18. Configuration Registry Server Data Stores (2 of 2)

Data Store	Type	Functionality
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.

4.8.1.6 Machine-to-Machine Gateway Server Software Description

4.8.1.6.1 Machine-to-Machine Gateway Server Functional Overview

The Machine-to-Machine Gateway (MTMGW) Server provides an automated ordering capability to allow the Science Investigator-Led Processing Systems (SIPS) to reprocess data externally from the ECS. In order to safeguard communications between the MTMGW Server and the SIPS, a SSH (Secure Shell protocol) is employed to secure the line. A SIPS user account is set up manually with SSH encryption keys on both the local SIPS host and a DAAC host. The SIPS Operations Staff initiates the key exchange using the SSH via a script.

The MTMGW Server is capable of receiving inventory search requests from the SIPS, submitting search requests to the DSS (SDSRV CSCI) for the selected metadata whose type information is obtained from the DMS (DDICT CSCI), and returning search results back to the SIPS.

The MTMGW Server is capable of accepting order requests based on UR or GranuleID from the SIPS, forwarding the corresponding acquire request to the Order Management Service (OMS) when configured to submit orders to the OMS or DSS (SDSRV CSCI) when configured to submit orders to SDSRV, and returning a response message to the SIPS indicating the status of order.

The MTMGW Server is capable of dealing with integrated search and order requests from the SIPS. In this case, no search result is returned to the SIPS. Instead, the search results are staged inside the MTMGW Server for further order processing. A response message is returned to the SIPS indicating the status of orders.

In the last two cases above, the ECS order tracking service keeps track of the MTMGW Server orders based on the user profile ID provided by the SIPS.

The MTMGW Server supports multiple concurrent servers; each server is independently configured by the DAAC. Each MTMGW Server supports multiple concurrent requests. DAAC operations can configure the maximum number of concurrent requests for each server.

4.8.1.6.2 Machine-to-Machine Gateway Server Context

Figure 4.8-12 is the Machine-to-Machine Gateway Server context diagram. Table 4.8-19 shows descriptions of the interface events in the Machine-to-Machine Gateway Server context diagram.

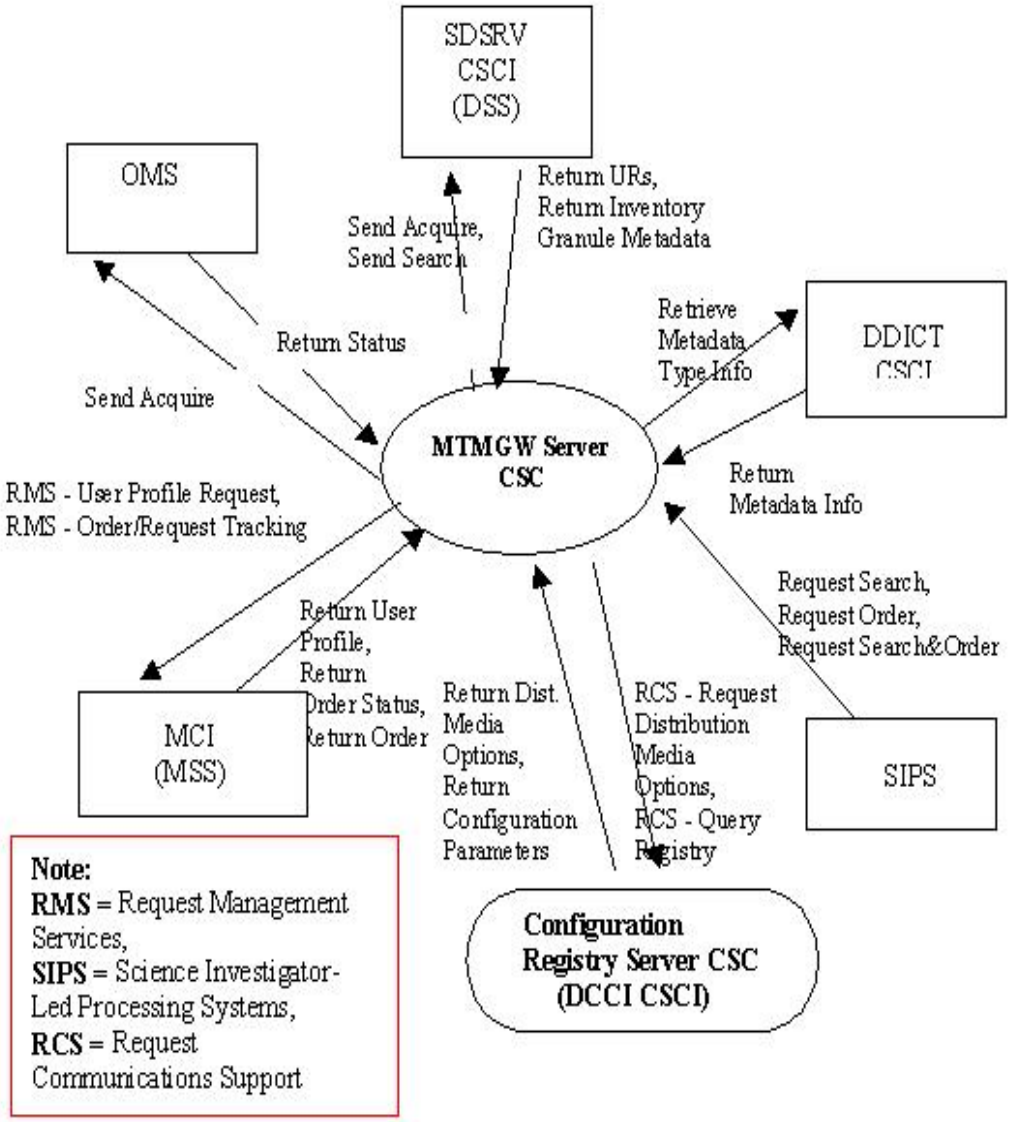


Figure 4.8-12. Machine-to-Machine Gateway Server Context Diagram

Table 4.8-19. Machine-to-Machine Gateway Server Interface Event (1 of 2)

Event	Interface Event Description
Send Acquire	The MTMGW supports a configuration parameter that specifies the manner of order submission to be either to the Order Management Service (OMS) or the ECS Science Data Server (SDSRV). When the submission parameter is configured to OMS, the MTMGW Server CSC submits acquire requests to the **OMS CI** for order or search & order requests from the SIPS, When the submission parameter is configured to SDSRV, the MTMGW Server CSC submits acquire requests to the **SDSRV CSCI** for order or search & order requests from the SIPS.
Send Search	The MTMGW Server CSC sends search requests to the **SDSRV CSCI** on behalf of the SIPS.
Retrieve Metadata Type Info	The MTMGW Server CSC retrieves metadata type information from the **DDICT CSCI** pertaining to search or search & order requests from the SIPS.
Return Metadata Info	The MTMGW Server CSC receives metadata information from the **DDICT CSCI**.
Request Search	The MTMGW Server CSC receives search requests from the **SIPS for ECS products**.
Request Order	The MTMGW Server CSC receives order requests from the **SIPS for ECS products**.
Request Search&Order	The MTMGW Server CSC receives integrated search and order requests from the **SIPS for ECS products**.
Request Communications Support (RCS)	The **DCCI CSCI** provides a library of services available to each SDPS and CSMS CSCI/CSC. The CSCI services required to perform specific assignments are requested from the DCCI CSCI/CSC. These services include: • CCS Middleware Support • Database Connection Services • File Transfer Services • Network & Distributed File Services • Bulk Data Transfer Services • Name/Address Services • Password Services • Server Request Framework (SRF) • Universal Reference (UR) • Error/Event Logging • Fault Handling Services • Mode Information • Query Registry - Retrieving the requested attribute-value pairs from the Configuration Registry • Request Distribution Media Options from the Configuration Registry
Return Dist. Media Options	The MTMGW Server CSC receives the distribution media options from the **Configuration Registry Server** CSC.
Return Configuration Parameters	The MTMGW Server CSC receives the requested configuration parameters from the **DCCI CSCI (Configuration Registry Server)**.

Table 4.8-19. Machine-to-Machine Gateway Server Interface Event (2 of 2)

Event	Interface Event Description
Request Management Services	The **MCI** 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 includes: • **System startup and shutdown -** Please refer to the release-related, current version of the Mission Operations Procedures for the ECS Project document (611) and the current ECS Project Training Material document (625), identified in Section 2.2.1 of this document. • **User Profile Request** – The **MCI** provides requesting CSCIs with User Profile parameters such as distribution priority and shipping address to support their processing activities. • **Order/Request Tracking** – The **MCI** provides an order tracking service to requesting CSCIs for creating and tracking order or search and order requests from the SIPS.
Return User Profile	The MTMGW Server CSC receives user profile information from the **MCI** to authenticate a user.
Return Order Status	The MTMGW Server CSC receives the status of an order for the requested ECS product from the **MCI**.
Return Order	The MTMGW Server CSC receives the product order object from the **MCI**.
Return URs	The MTMGW Server CSC receives Earth Science Data Type (ESDT) Universal References (URs) for the granules from the OMS or **SDSRV CSCI**.
Return Inventory Granule Metadata	The MTMGW Server CSC receives the inventory granule metadata identifying the scene within the granule from the **SDSRV CSCI** based on an inventory search request.

4.8.1.6.3 Machine-to-Machine Gateway Server Architecture

Figure 4.8-13 is the Machine-to-Machine Gateway Server architecture diagrams. The diagrams show the events sent to the Machine-to-Machine Gateway Server process and the events the Machine-to-Machine Gateway process sends to other processes.

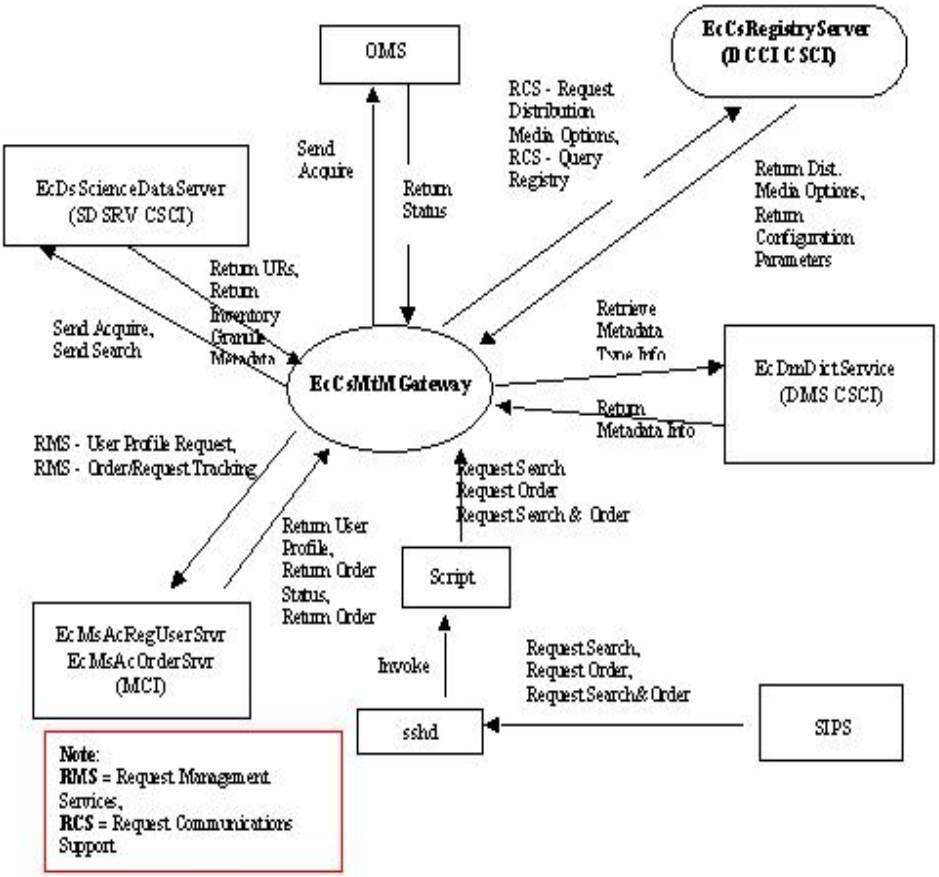


Figure 4.8-13. Machine-to-Machine Gateway Server Architecture Diagram

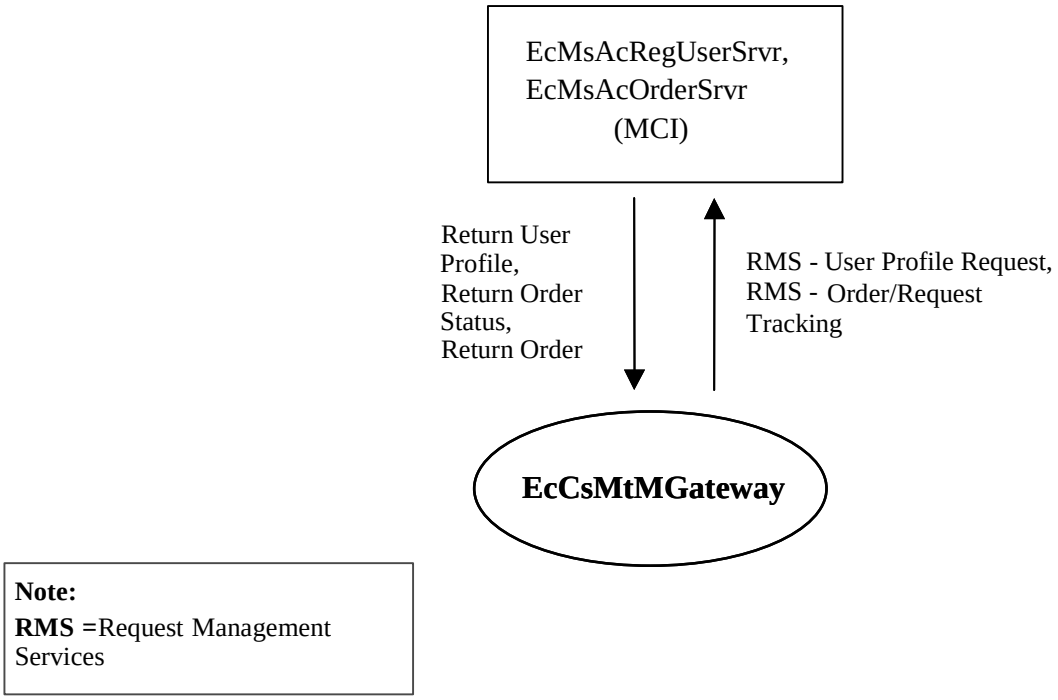


Figure 4.8-13. Machine-to-Machine Gateway Server Architecture Diagram (cont.)

4.8.1.6.4 Machine-to-Machine Gateway Server Process Descriptions

Table 4.8-20 provides a description of the processes in the Machine-to-Machine Gateway Server architecture diagrams.

Table 4.8-20. Machine-to-Machine Gateway Server Processes

Process	Type	COTS / Developed	Functionality
EcCsMtMGateway	Server	Developed	The Machine-to-Machine Gateway Server receives search, order, and integrated search & order requests from the SIPS via a sshd daemon. To fulfill search requests, the Machine- to-Machine Gateway Server uses the metadata type information retrieved from the EcDmDictServer process as the search request input to the EcDsScienceDataServer process and returns the search results back to the SIPS via the sshd. To process orders, the Machine-to-Machine Gateway Server process sends user profile requests to the EcMsAcRegUserSrvr process (using a user profile ID sent as part of the order request) to get distribution priority and shipping information from the user profile. The Machine-to-Machine Gateway Server process then sends order tracking requests to the EcMsAcOrderSrvr process to create an order, and submits acquire requests to the EcOmOrderManager when configured to submit orders to OMS or EcDsScienceDataServer when configured to submit orders to SDSRV. Finally, the Machine-to-Machine Gateway Server returns a response message back to the SIPS via the sshd. For search & order requests, the Machine-to-Machine Gateway Server process integrates the above two cases into one by staging the search results of the first step without returning to the SIPS and making order requests immediately.

4.8.1.6.5 Machine-to-Machine Gateway Server Process Interface Descriptions

Table 4.8-21 provides descriptions of the interface events shown in the Machine-to-Machine Gateway Server architecture diagram.

Table 4.8-21. Machine-to-Machine Gateway Server Process Interface Events (1 of 6)

Event	Event Frequency	Interface	Initiated by	Event Description
Request Communications Support	One service per request	***Process:*** EcCsIdNameServer ***Libraries:*** EcPf, Middleware, FoNs, FoIp, oodce *Classes:* EcPfManagedServer, CCSMdwNameServer, FoNsNameServerProxy, CCSMdwRwNetProxy ***Library (Common):*** EcUr *Class:* EcUrServerUR ***Library:*** event *Class:* EcLgErrorMsg ***Process:*** EcCsRegistry *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	***Process:*** EcCsMtMGateway *Library:* EcCsMtMGateway *Class:* EcCsMtMManagedSrv	The **DCCI CSCI** provides a library of services available to each SDPS and CSMS CSCI/CSC. The CSCI services required to perform specific assignments are requested from the DCCI CSCI. These services include: • CCS Middleware Support • Database Connection Services • File Transfer Services • Network & Distributed File Services • Bulk Data Transfer Services • Name/Address Services • Password Services • Server Request Framework (SRF) • Universal Reference (UR) • Error/Event Logging • Fault Handling Services • Mode Information • Query Registry - Retrieving the requested configuration attribute-value pairs from the Configuration Registry • Request Distribution Media Options from the Configuration Registry

Table 4.8-21. Machine-to-Machine Gateway Server Process Interface Events (2 of 6)

Event	Event Frequency	Interface	Initiated by	Event Description
Retrieve metadata type info	One or more client request	*Processes:* EcCsMtMGateway, EcDmDictServer *Libraries:* DmAsGwCommon, Common, DmGwV0Util, EcDmDClient, DmDdMsg *Class:* DmDdClRequestServer	*Process:* EcCsMtMGateway *Class:* EcCsMtMAttributeDict	The EcCsMtMGateway retrieves metadata type information from the **EcDmDictServer** based upon the qualifying metadata contained in the requests sent by the SIPS via a CCS Middleware call.
Return Metadata Info	Per search or order request	*Process:* EcCsMtMGateway *Class:* EcCsMtMAttributeDict	*Processes:* EcCsMtMGateway, EcDmDictServer *Libraries:* DmAsGwCommon, Common, DmGwV0Util, EcDmDClient, DmDdMsg *Class:* DmDdClRequestServer	The EcCsMtMGateway receives metadata information from the **EcDmDictServer**.
Request Search	One per client	*Process:* EcCsMtMGateway sshd *Script:* mtmsearch	*Process:* sshd (COTS)	The SIPS send search requests to the sshd. The **sshd** decrypts the request and forwards it to the EcCsMtMGateway Server.
Request Order	One per client	*Process:* EcCsMtMGateway sshd *Script:* mtmorder	*Process:* sshd (COTS)	The SIPS send order requests to the sshd. The **sshd** decrypts the request and forwards the request to the EcCsMtMGateway Server.
Request Search & Order	One per client	*Process:* EcCsMtMGateway sshd *Script:* mtmsearchorder	*Process:* sshd (COTS)	The SIPS send search & order requests to the sshd. The **sshd** decrypts the request and forwards the request to the EcCsMtMGateway Server.

Table 4.8-21. Machine-to-Machine Gateway Server Process Interface Events (3 of 6)

Event	Event Frequency	Interface	Initiated by	Event Description
Invoke Command	One or more per client	*Process:* sshd *Scripts:* mtmsearch, mtmorder, mtmsearchorder	Process: sshd (COTS)	The sshd daemon process invokes the sshd command on the SIPS side to send an invoke command to start the scripts on the ECS side of the interface to accept and process requests.
Send Acquire	One or more client request	*Process:* *EcOmOrderManager or*EcDsScienceDataServer *Libraries:* DsCl, DsCn, DsSh, DsGe, DsSr, DsDe2, Gl *Class:* DsClESDTReferenceCollector	*Process:* EcCsMtMGateway *Classes:* EcCsMtMDataServerMgr, EcCsMtMECSOrderProxy, EcCsMtMOrderImp, EcCsMtMECSSearchOrderProxy, EcCsMtMSearchOrderImp	The EcCsMtMGateway Server sends acquire requests based upon the created order constructed with the qualifying metadata contained in requests received from the SIPS to the EcOmOrderManager when configured the order submission to OMS or **EcDsScienceDataServer** via CCS Middleware calls when configured the order submission to SDSRV.
Send Search	One or more client request	*Process:* EcDsScienceDataServer *Libraries:* DsCl, DsCn, DsSh, DsGe, DsSr, DsDe2, Gl *Class:* DsClESDTReferenceCollector	*Process:* EcCsMtMGateway *Class:* EcCsMtMECSSearchProxy, EcCsMtMSearchImp, EcCsMtMSdsrvMgr, EcCsMtMDataServerMgr	The EcCsMtMGateway sends search requests, constructed based upon the qualifying metadata information received from the SIPS requests, to the **EcDsScienceDataServer** via CCS Middleware calls for inventory searches.

Table 4.8-21. Machine-to-Machine Gateway Server Process Interface Events (4 of 6)

Event	Event Frequency	Interface	Initiated by	Event Description
Return URs	One or more client request	*Process:* EcCsMtMGateway *Classes:* EcCsMtMDataServerMgr, EcCsMtMECSOrderProxy, EcCsMtMOrderImp, EcCsMtMECSSearchOrderProxy, EcCsMtMSearchOrderImp	*Process:* EcDsScienceDataServer *Libraries:* DsCl, DsSh *Classes:* DsClRequest, DsClCommand, DsClESDTReferenceCollector	The EcCsMtMGateway receives Earth Science Data Type (ESDT) Universal References (URs) for the granules from the **EcDsScienceDataServer**.
Return Inventory Granule Metadata	One or more client request	*Process:* EcCsMtMGateway *Class:* EcCsMtMECSSearchProxy, EcCsMtMSearchImp, EcCsMtMSdsrvMgr, EcCsMtMDataServerMgr	*Process:* EcDsScienceDataServer *Libraries:* DsCl, DsSh *Classes:* DsClESDTReference, DsClESDTReferenceCollector	The EcCsMtMGateway receives the inventory granule metadata identifying the scene within the granule based on an inventory search request sent to the **EcDsScienceDataServer**.
Return Dist. Media Options	One set of options per request	*Process:* EcCsMtMGateway *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	*Process:* EcCsRegistry *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	The EcCsMtMGateway receives the distribution media options from the **EcCsRegistry**.
Return Configuration Parameters	One per configuration registry query	Process: EcCsMtMGateway Library: EcCsRegistry Class: EcRgRegistryServer_C	Process: EcCsRegistry *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	The **EcCsRegistry** returns the requested configuration parameters to the EcCsMtMGateway.

Table 4.8-21. Machine-to-Machine Gateway Server Process Interface Events (5 of 6)

Event	Event Frequency	Interface	Initiated by	Event Description
Request Management Services (RMS)	One per service request	N/A	N/A	The **EcMsAcRegUserSrvr** and **EcMsAcOrderSrvr** 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 includes the items below.
RMS (cont.)	At system startup or shutdown and for restarts	*Process:* EcCsMtMGateway	DAAC unique startup scripts	**System startup and shutdown -** Please refer to the release-related, current version of the Mission Operations Procedures for the ECS Project document (611) and the current ECS Project Training Material document (625), identified in Section 2.2.1 of this document.
RMS (cont.)	One or more per client	*Processes:* EcCsMtMGateway, EcMsAcRegUserSrvr *Libraries:* MsAcClnt, MsAcComm	*Process:* EcCsMtMGateway *Class:* EcCsMtMAcctSrvMgr	**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.
RMS (cont.)	One or more per client	*Processes:* EcCsMtMGateway, EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	*Process:* EcCsMtMGateway *Class:* EcCsMtMAcctSrvMgr	**Order/Request Tracking** - The MCI provides an order tracking service (via the **EcMsAcOrderSrvr**) to requesting CSCIs for creating or tracking order or search and order requests from the SIPS.

Table 4.8-21. Machine-to-Machine Gateway Server Process Interface Events (6 of 6)

Event	Event Frequency	Interface	Initiated by	Event Description
Return User Profile	One per user request	*Process:* EcCsMtMGateway *Class:* EcCsMtMAcctSrvMgr	*Processes:* EcCsMtMGateway, EcMsAcRegUserSrvr *Libraries:* MsAcClnt, MsAcComm	The EcCsMtMGateway receives user profile information from the **EcMsAcRegUserSrvr**.
Return Order Status	One per user request	*Process:* EcCsMtMGateway *Class:* EcCsMtMAcctSrvMgr	*Processes:* EcCsMtMGateway, EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	The EcCsMtMGateway receives the status of an order from the **EcMsAcOrderSrvr**.
Return Order	One per user request	*Process:* EcCsMtMGateway *Class:* EcCsMtMAcctSrvMgr	*Processes:* EcCsMtMGateway, EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	The EcCsMtMGateway receives the product order object from the **EcMsAcOrderSrvr**.

4.8.1.6.6 Machine-to-Machine Gateway Server Data stores

Data stores are not applicable for the Machine-to-Machine Gateway Server.

4.8.1.7 CCS Middleware Support Group Description

The CCS Middleware support group consists of the CCS Name Server.

4.8.1.7.1 CCS Middleware Functional Overview

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

4.8.1.7.2 CCS Middleware Context

Figure 4.8-14 is the CCS Middleware context diagram. Table 4.8-22 provides descriptions of the interface events shown in the CCS Middleware context diagram.

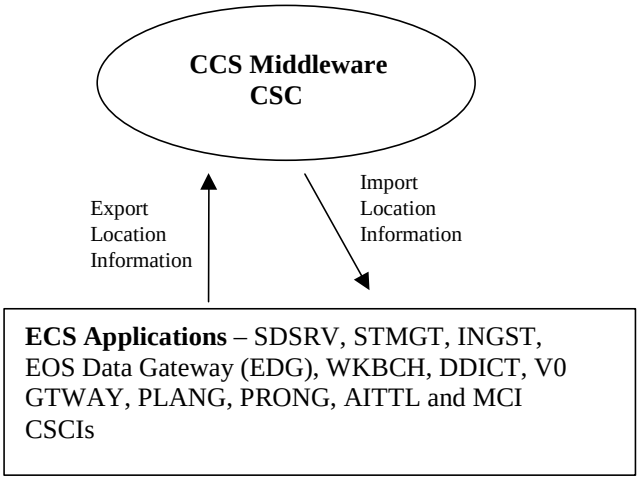


Figure 4.8-14. CCS Middleware Context Diagram

Table 4.8-22. CCS Middleware Interface Events

Event	Interface Event Description
Import location information (binding information)	An **ECS application** requests server location information from the CCS Name Server and saves the information in its local cache via the CCS Name Server client proxy component.
Export location information (binding information)	The CCS Middleware CSC stores physical and logical location information received from **ECS Applications** in the CCS Name Server via the CCS Name Server client proxy component.

4.8.1.7.3 CCS Middleware Architecture

Figure 4.8-15 is the CCS Middleware support group architecture diagram. The diagram shows the events sent to the CCS Middleware process and the events the CCS Middleware process send to other processes.

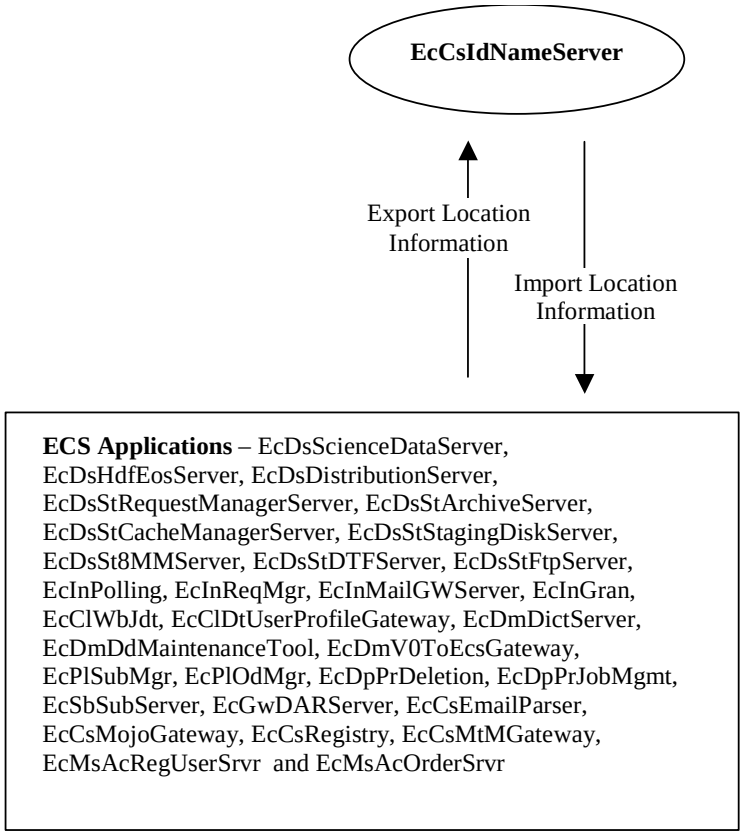


Figure 4.8-15. CCS Middleware Architecture Diagram

4.8.1.7.4 CCS Middleware Process Descriptions

Table 4.8-23 provides descriptions of the processes shown in the CCS Middleware architecture diagram. ECS applications store server location information on the CCS Name Server (EcCsIdNameServer).

Table 4.8-23. CCS Middleware Processes

Process	Type	COTS/ Developed	Functionality
EcCsIdNameServer	Internal	COTS	Stores server location information and provides interfaces for storing and retrieving the location information.

4.8.1.7.5 CCS Middleware Process Interface Descriptions

Table 4.8-24 provides descriptions of the interface events shown in the CCS Middleware architecture diagram.

Table 4.8-24. CCS Middleware Process Interface Events

Event	Event Frequency	Interface	Initiated By	Event Description
Import location information	One per server	*Process:* EcCsIdNameServer *Libraries:* EcPf, Middleware, FoNs, FoIp, oodce *Classes:* EcPfManagedServer, CCSMdwNameServer, FoNsNameServerProxy, CCSMdwRwNetProxy	*Processes:* ECS Applications (All servers identified in the architecture diagram.) *Libraries:* EcPf, Middleware, FoNs, FoIp, oodce *Classes:* EcPfManagedServer, CCSMdwNameServer, FoNsNameServerProxy, CCSMdwRwNetProxy	An **ECS application** retrieves server location information from the EcCsIdNameServer via the CCS Name Server client component in the ECS application.
Export location information	One per server	*Process:* EcCsIdNameServer *Libraries:* EcPf, Middleware, FoNs, FoIp, oodce *Classes:* EcPfManagedServer, CCSMdwNameServer, FoNsNameServerProxy, CCSMdwRwNetProxy	*Processes:* ECS Applications (All processes identified in the architecture diagram.) *Libraries:* EcPf, Middleware, FoNs, FoIp, oodce *Classes:* EcPfManagedServer, CCSMdwNameServer, FoNsNameServerProxy, CCSMdwRwNetProxy	The **ECS application** places physical and logical location information in the EcCsIdNameServer via the CCS Name Server client component in the ECS application.

4.8.1.7.6 CCS Middleware Data Stores

Table 4.8-25 provides a description of the data store shown in the CCS Middleware architecture diagram.

Table 4.8-25. CCS Middleware Data Stores

Data Store	Type	Functionality
StringId	Sybase	This data store contains the ECS Mode information.
Service	Sybase	This data store contains the service name an ECS application server listens on.
Host	Sybase	This data store contains the host name an ECS application server runs on.
ProcessId	Sybase	This data store contains the process id an ECS application server runs with.
ClassId	Sybase	This data store contains the binding information of an ECS application server, which includes process id and port number.
ServerUR.map	Other	A flat file for the Server Locator classes to map short, logical service names to CCS Name Server entry names.

4.8.1.8 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.

4.8.1.8.1 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.

4.8.1.8.2 File Transfer Protocol Context

Figure 4.8-16 is the FTP context diagram. Table 4.8.1.8.2-1 provides descriptions of the interface events in the FTP context diagram.

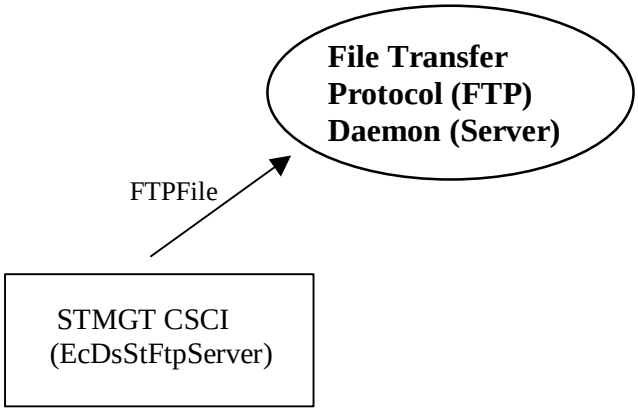


Figure 4.8-16. File Transfer Protocol Server Context Diagram

Table 4.8-26. File Transfer Protocol Interface Events

Event	Interface Event Description
FTPFile	The **STMGT CSCI** sends requests to the FTP Daemon to transfer the files to the Pull cache or to an external user.

4.8.1.8.3 File Transfer Protocol Architecture

Figure 4.8-17 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.

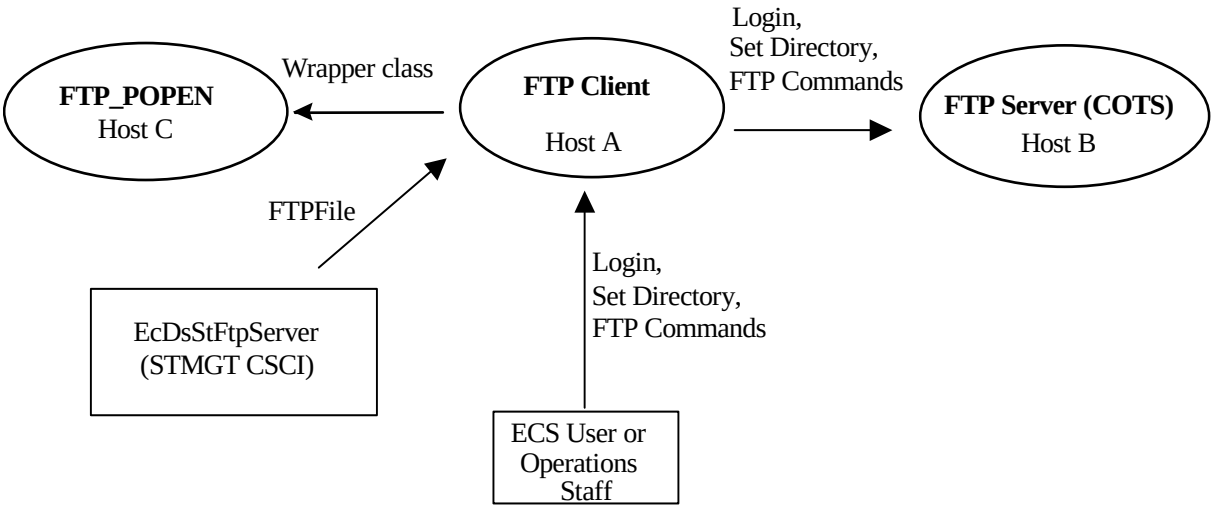


Figure 4.8-17. File Transfer Protocol Architecture Diagram

4.8.1.8.4 File Transfer Protocol Process Descriptions

Table 4.8-27 provides descriptions of the processes shown in the File Transfer Protocol architecture diagram.

Table 4.8-27. 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_POPEN)	Client Application	Developed	**ECS application processes** use the Wrapper class whenever FTP is used with other class methods.

4.8.1.8.5 File Transfer Protocol Process Interface Descriptions

Table 4.8-28 provides descriptions of the interface events shown in the File Transfer Protocol architecture diagram.

Table 4.8-28. File Transfer Protocol Process Interface Events (1 of 2)

Event	Event Frequency	Interface	Initiated by	Event Description
Login	One per connection to FTP	*Process:* FTP Server Daemon *Library:* CsFtFTPRelA *Class:* CsFtFTPRelA	User/Operations Staff *Process:* FTP Client *Classes:* DsStFtpService, 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	*Process:* FTP Server *Library:* CsFtFTPRelA *Class:* CsFtFTPRelA	User/Operations Staff *Process:* FTP Client *Classes:* DsStFtpService, EcMjECSFtpProxy, 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	*Process:* FTP Server *Library:* CsFtFTPRelA *Class:* CsFtFTPRelA	User/Operations Staff *Process:* FTP Client *Classes:* DsStFtpService, EcMjECSFtpProxy, 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.
FTPFile	One per file	*Process:* FTP Client *Classes:* DsStFtpService, InMessage	*Process:* EcDsStFtpServer	The EcDsStFtpServer sends a file transfer command to the FTP Client to have a file transferred to a directory on the same or another host.

Table 4.8-28. File Transfer Protocol Process Interface Events (2 of 2)

Event	Event Frequency	Interface	Initiated by	Event Description
Wrapper class	One per FTP	*Process:* FTP_POPEN *Library:* CsFtFTPRelA *Class:* CsFtFTPRelA	*Process:* FTP Client *Classes:* DsStFtpService, EcMjECSFtpProxy, InMediaIngestRPUtil, InMessage, DpPrDataManager	The **FTP Client** provides wrapper functions, to the FTP_POPEN, to carry out FTP between two hosts.

4.8.1.8.6 File Transfer Protocol Data Stores

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

4.8.1.9 Remote File Access Group - File Transfer Protocol Notification

4.8.1.9.1 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 **(SetFtpDoneFiles)** 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.

4.8.1.9.2 File Transfer Protocol Notification Context

Figure 4.8-18 is the File Transfer Protocol Notification context diagram. Table 4.8-29 provides descriptions of the interface events shown in the File Transfer Protocol Notification context diagram.

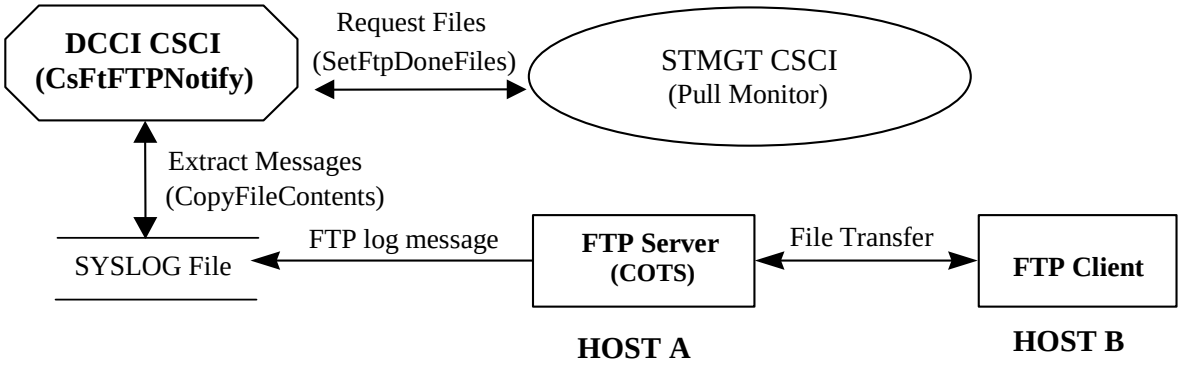


Figure 4.8-18. File Transfer Protocol Notification Context Diagram

Table 4.8-29. File Transfer Protocol Notification Interface Events

Event	Interface Event Description
File Transfer	The file transfer of specified files between the FTP client and FTP server.
FTP log message	The FTP server logs all FTP events to the system log (syslog) file.
Extract messages	The "CopyFileContents" 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.

4.8.1.9.3 File Transfer Protocol Notification Architecture

Figure 4.8-19 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.

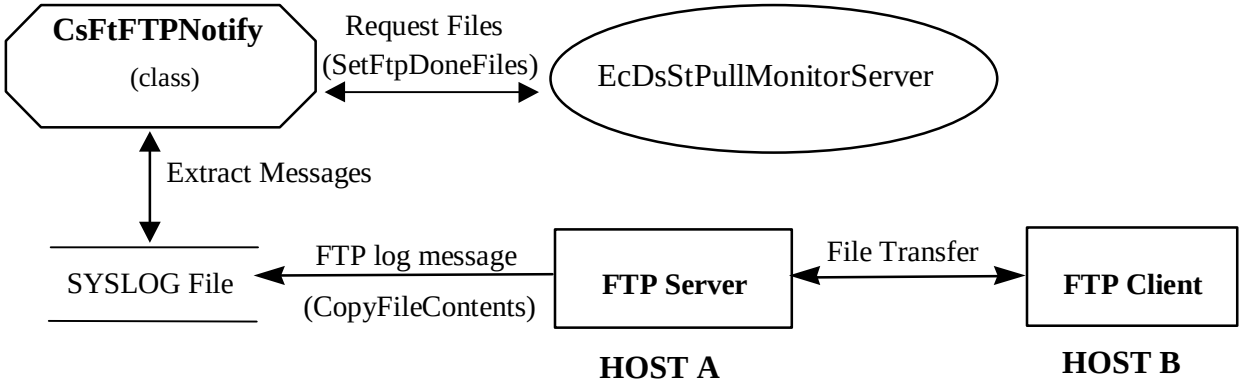


Figure 4.8-19. File Transfer Protocol Notification Architecture Diagram

4.8.1.9.4 File Transfer Protocol Notification Process Descriptions

Table 4.8-30 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 4.8-30. 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. The CsFtFTPNotify class makes a copy of the logs.
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.

4.8.1.9.5 File Transfer Protocol Notification Process Interface Descriptions

Table 4.8-31 provides descriptions of the interface events shown in the File Transfer Protocol Notification architecture diagram.

Table 4.8-31. File Transfer Protocol Notification Process Interface Events

Event	Event Frequency	Interface	Initiated by	Event Description
File Transfer	One per file transfer	*Process:* FTP Server	*Process:* FTP Client Application	The specified file transfer takes place between the FTP client and FTP server.
FTP log message	One per system log file entry	SYSLOG File	*Process:* 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	*Process:* EcDsStPullMonitor Server *Library:* CsFtFTPNotify *Class:* CsFtFTPNotify	The **EcDsStPullMonitorServer**, via the **CsFtFTPNotify** class is used to extract messages from the system log (syslog) file.
Request Files	One per FTP request	*Function:* SetFtpDoneFiles *Library:* CsFtFTPNotify *Class:* CsFtFTPNotify	*Process:* EcDsStPullMonitor Server *Class:* 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.

4.8.1.9.6 File Transfer Protocol Notification Data Stores

Table 4.8-32 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 the CDRL 311, 311-CD-626).

Table 4.8-32. File Transfer Protocol Data Stores

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

4.8.1.10 Remote File Access Group - Bulk Data Server Description

4.8.1.10.1 Bulk Data Server Functional Overview

The Bulk Data Server (BDSpro product) 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. BDSpro is available only on SGI IRIX platforms. Figure 4.8-20 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. BDSpro exploits the data access speed of the Extended File System (XFS) and data transfer rates of network media, such as high-speed Gig Ethernet 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 6.5.6 (or later versions) and connected to a high-speed network (such as high-speed Gig Ethernet or fiberchannel) running the TCP/IP suite.

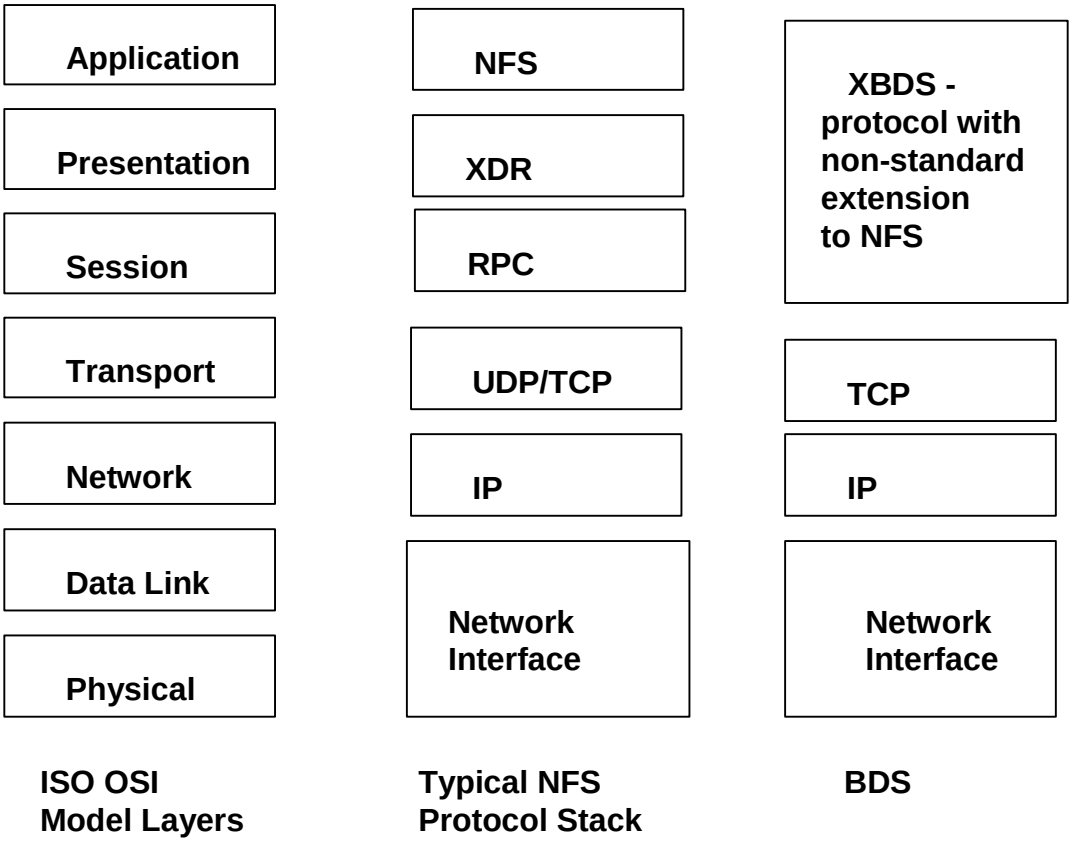


Figure 4.8-20. Bulk Data Server Protocol compared with ONC Protocols

4.8.1.10.2 Bulk Data Server Context

BDS is a file transfer utility to move large data files over the high-speed Gig Ethernet communications lines. Figure 4.8-21 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 the BDS protocol over high-speed Gig Ethernet and stored on AMASS.

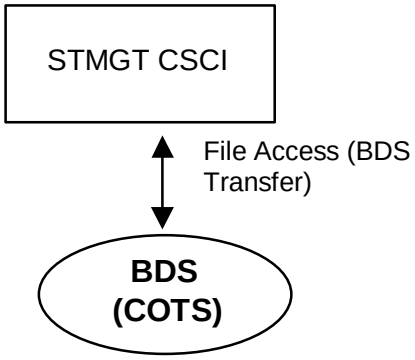


Figure 4.8-21. Bulk Data Server Context Diagram

Table 4.8-33 provides a description of the interface event shown in the Bulk Data Server context diagram.

Table 4.8-33. Bulk Data Server Interface Events

Event	Interface Event Description
File Access (BDS transfer)	The **STMGT CSCI** uses the BDS to transfer large data files over the high-speed Gig Ethernet interface for storage in the Science Data Server archives.

4.8.1.10.3 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 4.8-22 is the BDS architecture diagram shown over high-speed Gig Ethernet on an SGI client-server model.

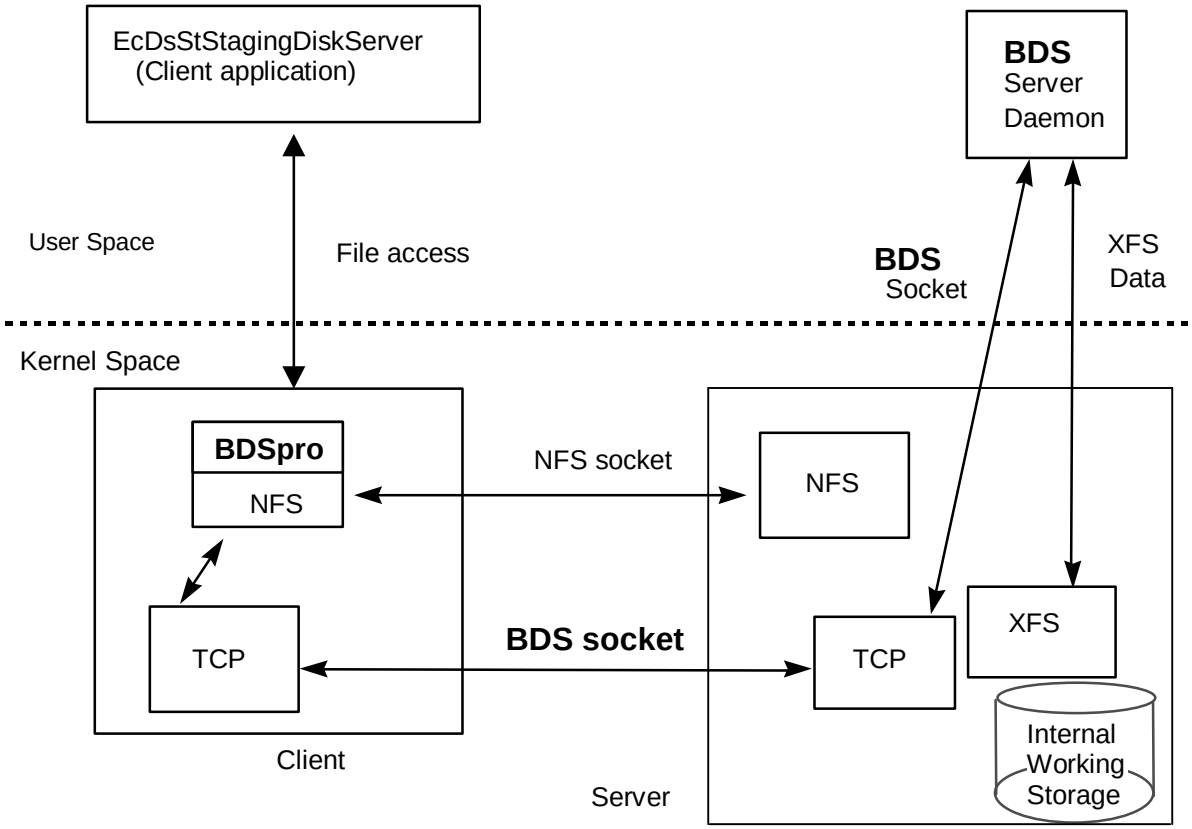


Figure 4.8-22. Bulk Data Server Architecture Diagram

4.8.1.10.4 Bulk Data Server Process Descriptions

Table 4.8-34 provides descriptions of the processes shown in the Bulk Data Server architecture diagram.

Table 4.8-34. 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.

4.8.1.10.5 Bulk Data Server Process Interface Descriptions

Table 4.8-35 provides descriptions of the interface event shown in the Bulk Data Server architecture diagram.

Table 4.8-35. Bulk Data Server Process Interface Events

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

4.8.1.10.6 Bulk Data Server Data Stores

Data stores are not applicable for the Bulk Data Server.

4.8.1.11 Remote File Access Group - Network File System Description

4.8.1.11.1 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.

4.8.1.11.2 Network File System Context

Figure 4.8-23 is the Network File System context diagram. The NFS mounted directories reside on mount points made accessible for the use of other host machines. Table 4.8-36 provides descriptions of the interface event shown in the context diagram.

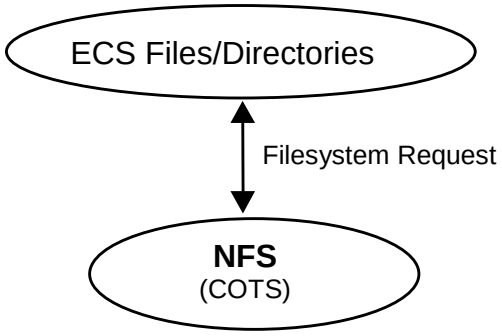


Figure 4.8-23. Network File System Context Diagram

Table 4.8-36. Network File System Interface Events

Event	Interface Event Description
Filesystem Request	The NFS clients request **ECS 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.

4.8.1.11.3 Network File System Architecture

Figure 4.8-24 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.

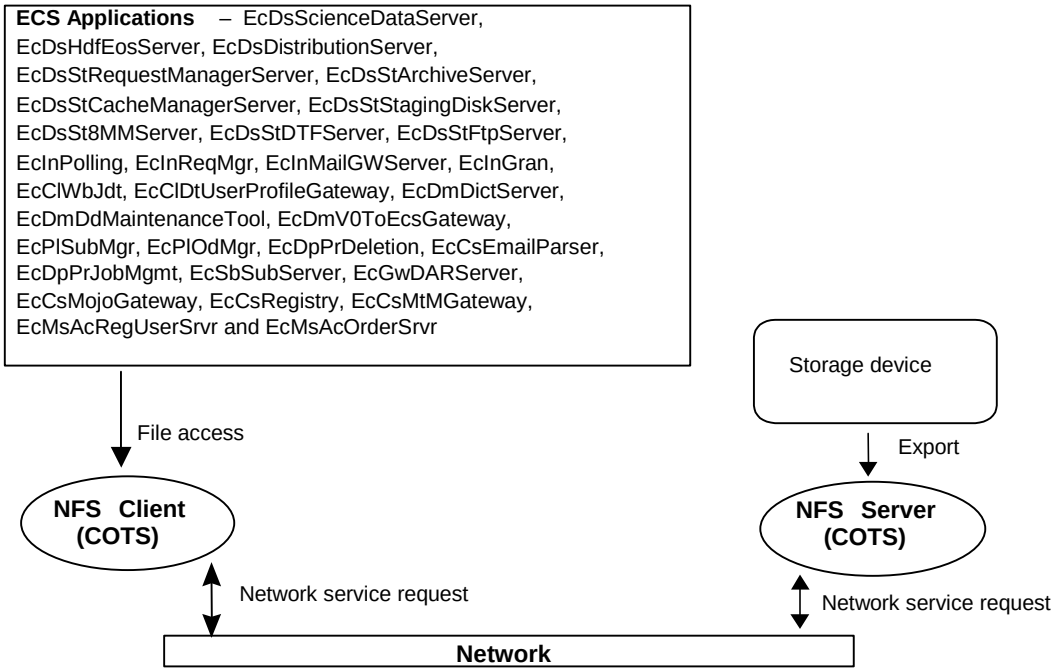


Figure 4.8-24. Network File System Architecture Diagram

4.8.1.11.4 Network File System Process Descriptions

Table 4.8-37 provides descriptions of the processes shown in the Network File System architecture diagram.

Table 4.8-37. 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.

4.8.1.11.5 Network File System Process Interface Descriptions

Table 4.8-38 provides the descriptions of the interface events shown in the NFS architecture diagram.

Table 4.8-38. Network File System Process Interface Events

Event	Event Frequency	Interface	Initiated by	Event Description
File Access	One per file access	NFS Client (COTS)	*Processes:* EcDsScienceDataServer, EcDsHdfEosServer, EcDsDistributionServer, EcDsStRequestManagerServer, EcDsStArchiveServer, EcDsStCacheManagerServer, EcDsStStagingDiskServer, EcDsSt8MMServer, EcDsStDTFServer, EcDsStFtpServer, EcInPolling, EcInReqMgr, EcInMailGWServer, EcInGran, EcClWbJdt, EcClDtUserProfileGateway, EcDmDictServer, EcDmDdMaintenanceTool, EcDmV0ToEcsGateway, EcPlSubMgr, EcPlOdMgr, EcDpPrDeletion, EcDpPrJobMgmt, EcSbSubServer, EcGwDARServer, EcCsEmailParser, EcCsMojoGateway, EcCsRegistry, EcCsMtMGateway, EcMsAcRegUserSrvr, EcMsAcOrderSrvr	The **ECS application** uses the file accessed via the NFS client.
Export	One per server export	NFS Server (COTS)	Storage Device	The NFS server exports the details of **storage devices** to verify clients.

4.8.1.11.6 Network File System Data Stores

Data stores are not applicable for the Network File System.

4.8.1.12 Remote File Access Group - Filecopy Description

4.8.1.12.1 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.

4.8.1.12.2 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 Metadata Configuration Files (MCFs) are copied from the SDSRV to the DDIST staging Disk using Filecopy. Figure 4.8-25 is the Filecopy context diagram.

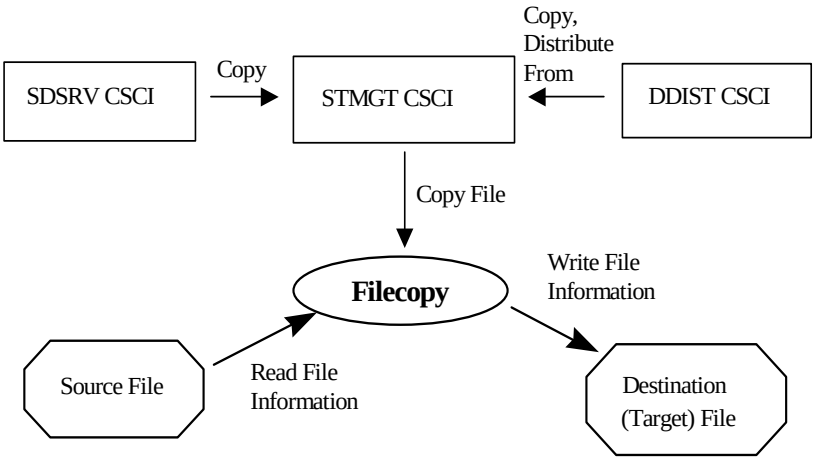


Figure 4.8-25. Filecopy Context Diagram

Table 4.8-39 provides descriptions of the interface events shown in the Filecopy context diagram.

Table 4.8-39. 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.
Read File Information	The data file information is read from the source file.
Write File Information	The data file information is written to the destination file.

4.8.1.12.3 Filecopy Architecture

The Filecopy utility uses options to provide copy features of file compression, decompression, or the standard copy. Figure 4.8-26 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.

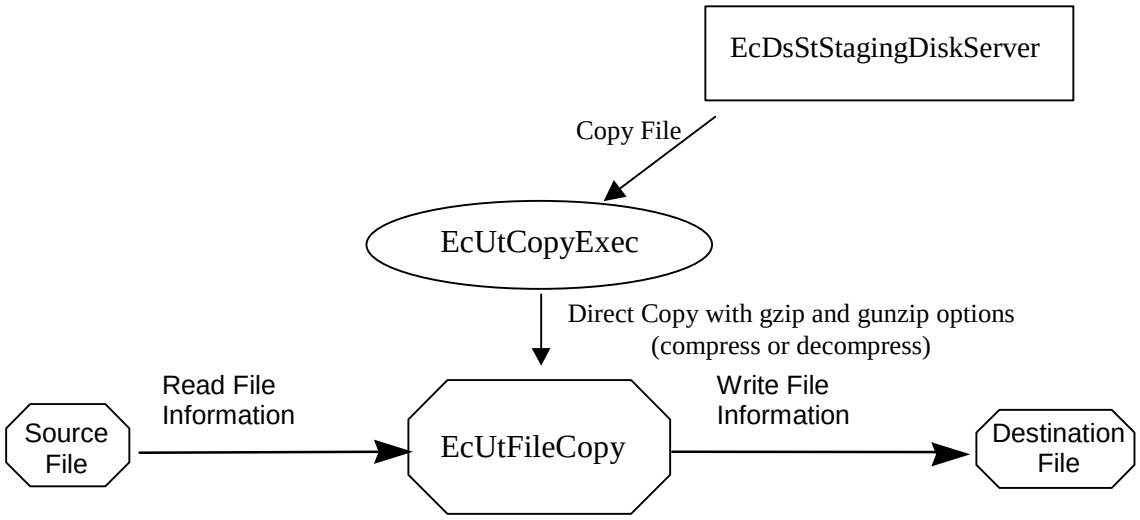


Figure 4.8-26. Filecopy Architecture Diagram

4.8.1.12.4 Filecopy Process Descriptions

Table 4.8-40 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 4.8-40. Filecopy Process

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

4.8.1.12.5 Filecopy Process Interface Descriptions

Table 4.8-41 provides descriptions of the interface events shown in the FileCopy architecture diagram.

Table 4.8-41. Filecopy Process Interface Events

Event	Event Frequency	Interface	Initiated by	Event Description
Copy File	One file copy per request	*Process:* EcUtCopyExec *Class:* EcUtCopyExec	*Process:* EcDsStStagingDiskServer *Libraries:* DsStSdClientAsync, DsStSdClientSync *Class:* 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	*Binary:* EcUtFilecopy main function	*Process:* EcUtCopyExec *Library:* EcUtMisc *Class:* EcUtCopyExec	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.
Read File Information	One per file copy	Source File	*Process:* EcUtCopyExec *Library:* EcUtMisc *Class:* EcUtCopyExec	Data file location information is read from the source file.
Write File Information	One per file copy	Destination File	*Process:* EcUtCopyExec *Library:* EcUtMisc *Class:* EcUtCopyExec	Data file location information is written to the destination file.

4.8.1.12.6 Filecopy Data Stores

Data stores are not applicable for the Filecopy service.

4.8.1.13 Mail Support Group Description

4.8.1.13.1 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.

4.8.1.13.2 E-mail Server Context

Figure 4.8-27 is the E-mail Server context diagram. Table 4.8-42 provides descriptions of the interface events shown in the E-mail Server context diagram.

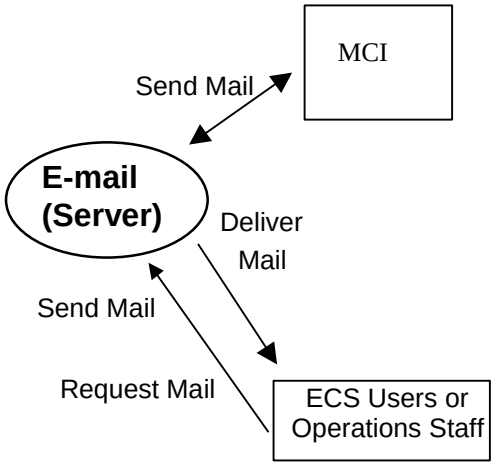


Figure 4.8-27. E-mail Server Context Diagram

Table 4.8-42. E-mail Server Interface Events

Event	Interface Event Description
Send Mail	The **MCI** uses the e-mail client software to send mail and the API spawns a *sendmail* process to deliver the message. Interactive users, use the COTS software product sendmail, which delivers the mail message.
Deliver Mail	The mail server delivers the mail to the addressed **ECS user** or **Operations Staff**.
Request Mail	**ECS users** or **Operations Staff** sends requests for e-mail messages to the E-mail Server.

4.8.1.13.3 E-mail Server Architecture

The E-mail server is a COTS software product. Figure 4.8-28 is the E-mail server architecture diagram.

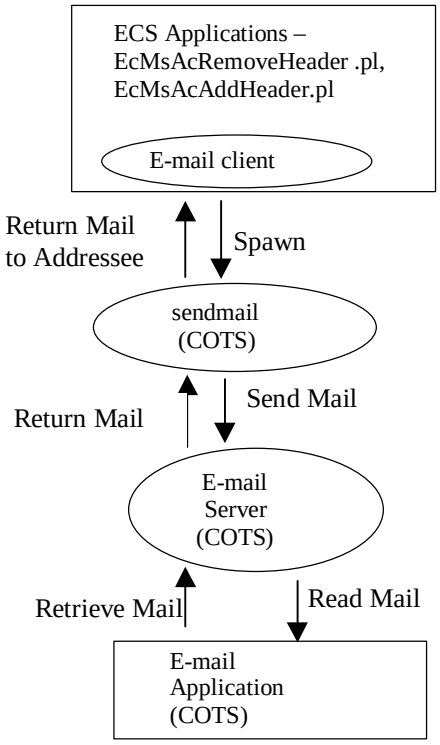


Figure 4.8-28. E-mail Server Architecture Diagram

4.8.1.13.4 E-mail Server Process Descriptions

Table 4.8-43 provides descriptions of the processes shown in the E-mail Server architecture diagram.

Table 4.8-43. 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 *sendmail* 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.

4.8.1.13.5 E-mail Server Process Interface Descriptions

Table 4.8-44 provides descriptions of the interface events shown in the E-mail server architecture diagram.

Table 4.8-44. E-mail Server Process Interface Events (1 of 2)

Event	Event Frequency	Interface	Initiated by	Event Description
Spawn	One per process spawned	*Library:* libc API: System ( )	*Scripts:* EcMsAcRemoveHeader.pl, EcMsAcAddHeader.pl *Library:* CsEmMailRelA *Class:* 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	*Process:* E-mail Server (COTS)	*Process:* 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	*Process:* E-mail Application (COTS)	*Process:* E-mail Server (COTS)	E-mail is received from the COTS application (sendmail) and routed to another user via the E-mail server.

Table 4.8-44. E-mail Server Process Interface Events (2 of 2)

Event	Event Frequency	Interface	Initiated by	Event Description
Retrieve mail	One per E-mail send	*Process:* E-mail Server (COTS)	*Process:* 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.
Return Mail	One per e-mail read	*Process:* Sendmail (COTS)	*Process:* E-mail Server (COTS)	E-mail is sent from the Sendmail package to a user via the E-mail Server.
Return Mail to Addressee	One per e-mail read	*Process:* E-mail Client	*Process:* Sendmail (COTS)	E-mail is retrieved by a user from the Sendmail package via the E-mail Server.

4.8.1.13.6 E-mail Server Data Stores

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

4.8.1.14 Virtual Terminal Description

4.8.1.14.1 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.

4.8.1.14.2 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 4.8-29 is the Virtual Terminal context diagram and Table 4.8-45 provides the descriptions of the interface events shown in the Virtual Terminal context diagram.

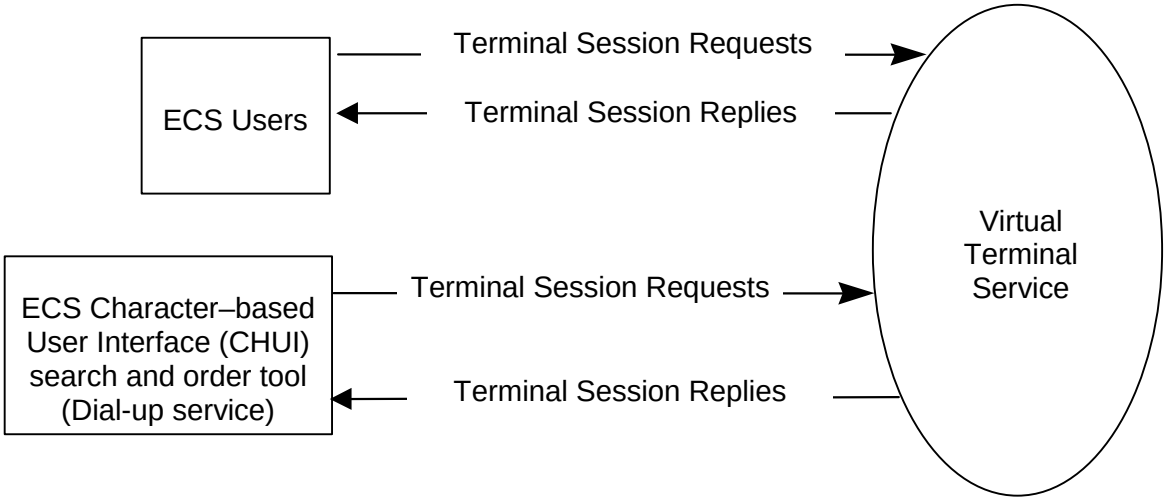


Figure 4.8-29. Virtual Terminal Context Diagram

Table 4.8-45. 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.

4.8.1.14.3 Virtual Terminal Architecture

Figure 4.8-30 is the Virtual Terminal architecture diagram. The diagram shows the event traffic between the Telnet with ECS Users and Telnet with Dial-up users.

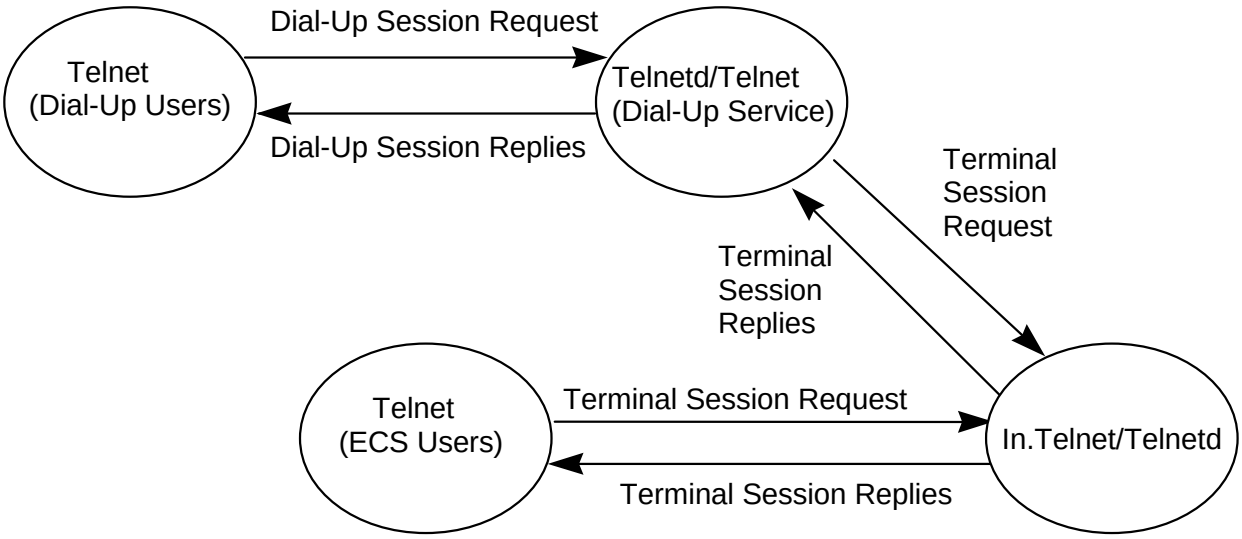


Figure 4.8-30. Virtual Terminal Architecture Diagram

4.8.1.14.4 Virtual Terminal Process Descriptions

Table 4.8-46 provides the descriptions of the processes shown in the Virtual terminal architecture diagram.

Table 4.8-46. 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.

4.8.1.14.5 Virtual Terminal Process Interface Descriptions

Table 4.8-47 provides the descriptions of the interface events shown in the Virtual Terminal architecture diagram.

Table 4.8-47. Virtual Terminal Process Interface Events

Event	Event Frequency	Interface	Initiated by	Event Description
Dial-up Session Request	One per connection request	*Process:* Telnetd/Telnet (dial-up service)	*Process:* 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	*Process:* Telnet (dial-up service)	*Process:* 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	*Process:* In.Telnet/Telnetd	*Process:* 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	*Process:* Telnet (ECS users)	*Process:* In.Telnet/Telnetd	The Host Virtual Terminal Process, Kerberos telnetd, responds to the connection requests and establishes or maintains the sessions.

4.8.1.14.6 Virtual Terminal Data Stores

Data stores are not applicable for the Virtual Terminal.

4.8.1.15 Cryptographic Management Interface Software Description

4.8.1.15.1 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 security required services (i.e., Sybase ASE). These passwords (and optionally login names) are generated dynamically based on a psuedo-random number used as the seed for the password.

4.8.1.15.2 Cryptographic Management Interface Context

Figure 4.8-31 is the Cryptographic Management Interface context diagram. Servers (PF or non-PF) use the CMI with a need for access to security required services. Table 4.8-48 provides descriptions of the interface events shown in the Cryptographic Management Interface context diagram.

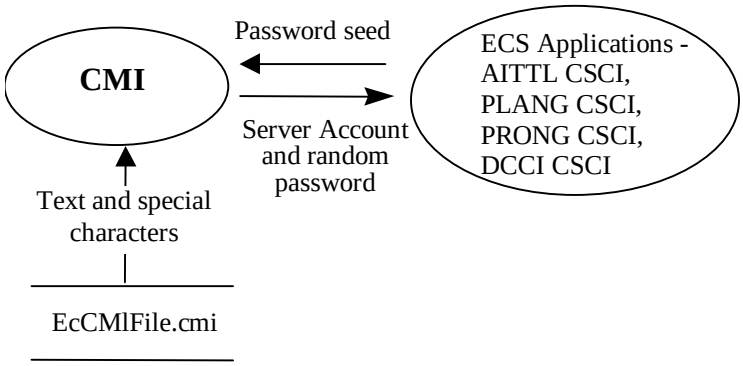


Figure 4.8-31. Cryptographic Management Interface Context Diagram

Table 4.8-48. Cryptographic Management Interface Events

Event	Interface Event Description
Password seed	The **ECS applications** (**AITTL**, **PLANG**, **PRONG and DCCI** CSCIs) request an account and provide 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.

4.8.1.15.3 Cryptographic Management Interface Architecture

Figure 4.8-32 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.

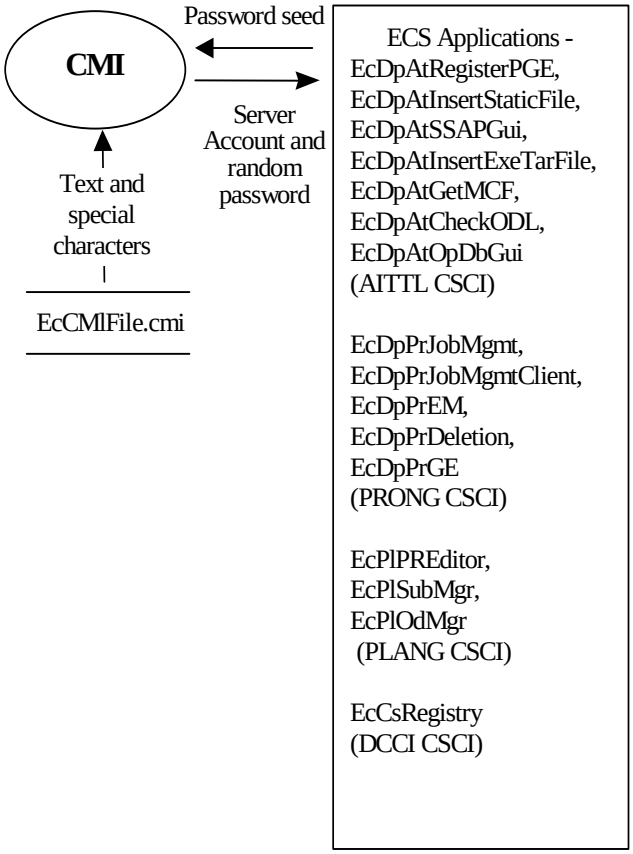


Figure 4.8-32. Cryptographic Management Interface Architecture Diagram

4.8.1.15.4 Cryptographic Management Interface Process Descriptions

Table 4.8-49 provides descriptions of the processes shown in the Cryptographic Management Interface context diagram.

Table 4.8-49. Cryptographic Management Interface Processes

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

4.8.1.15.5 Cryptographic Management Interface Process Interface Descriptions

Table 4.8-50 provides the descriptions of the interface events shown in the Cryptographic Management Interface architecture diagram.

Table 4.8-50. Cryptographic Management Interface Process Interface Events (1 of 2)

Event	Event Frequency	Interface	Initiated by	Event Description
Password seed	One per password seed	*Process:* CMI *Library:* EcSeCmi *Class:* EcSeCmi	***AITTL Processes:*** EcDpAtRegisterPGE, EcDpAtInsertStaticFile, EcDpAtSSAPGui, EcDpAtInsertExeTarFile, EcDpAtGetMCF, EcDpAtCheckODL, EcDpAtOpDbGui ***PLANG Processes:*** EcPlPREditor, EcPlSubMgr, EcPlOdMgr ***PRONG Processes:*** EcDpPrJobMgmt, EcDpPrJobMgmtClient, EcDpPrEM, EcDpPrDeletion, EcDpPrGE ***DCCI Process:*** EcCsRegistry	The server provides a unique number as a seed for generating a password to the **ECS Applications**.

Table 4.8-50. Cryptographic Management Interface Process Interface Events (2 of 2)

Event	Event Frequency	Interface	Initiated by	Event Description
Server Account and random password	One per account and password	***AITTL Processes:*** EcDpAtRegisterPGE, EcDpAtInsertStaticFile, EcDpAtSSAPGui, EcDpAtInsertExeTarFile, EcDpAtGetMCF, EcDpAtCheckODL, EcDpAtOpDbGui ***PLANG Processes:*** EcPlPREditor, EcPlSubMgr, EcPlOdMgr ***PRONG Processes:*** EcDpPrJobMgmt, EcDpPrJobMgmtClient, EcDpPrEM, EcDpPrDeletion, EcDpPrGE ***DCCI Process:*** EcCsRegistry	*Process:* CMI *Library:* EcSeCmi *Class:* EcSeCmi	CMI generates a random password for the account based on the seed.

4.8.1.15.6 Cryptographic Management Interface Data Stores

Table 4.8-51 provides descriptions of the data store shown in the Cryptographic Management Interface architecture diagram.

Table 4.8-51. Cryptographic Management Interface Data Stores

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

4.8.1.16 Domain Name Server Software Description

4.8.1.16.1 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 4.8-33 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.

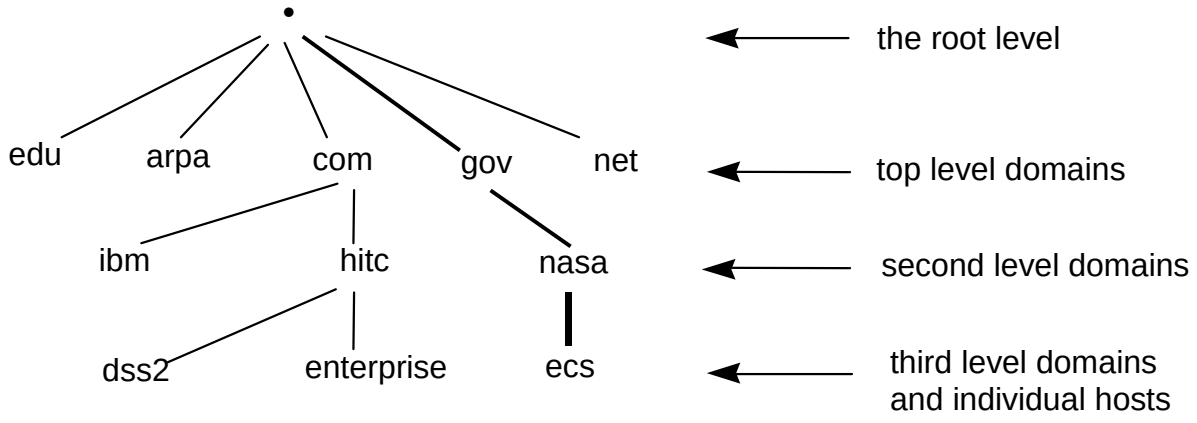


Figure 4.8-33. Domains Hierarchy Diagram

The fourth level domains in the ECS project include domains of DAACs: gsfcb, gsfcmo, and etc. Figure 4.8-34 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 production network and the latter are for the GSFC M&O network.

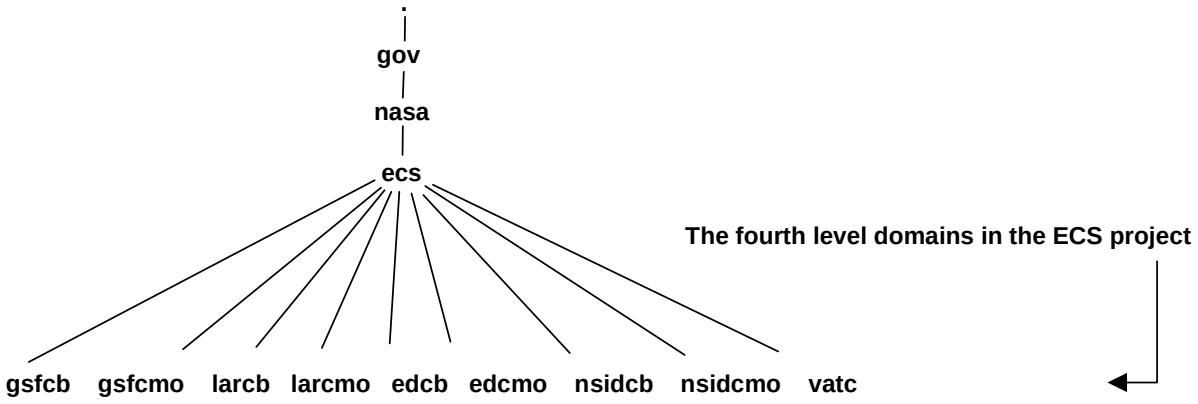


Figure 4.8-34. DNS Domains of the ECS Project Diagram

Figure 4.8-35 is the ECS topology domain diagram.

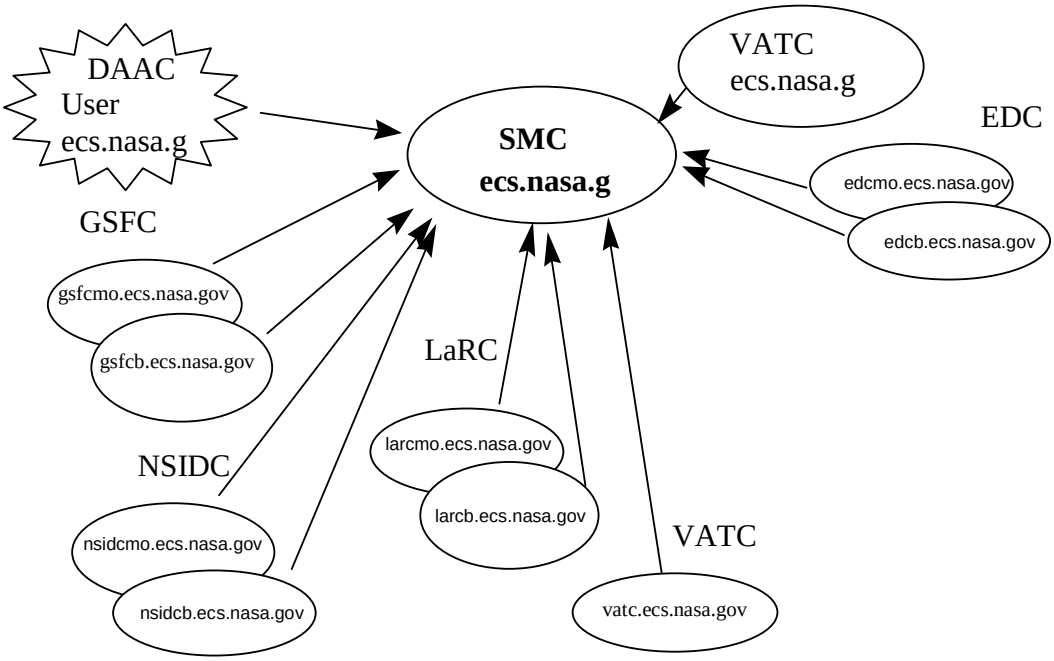


Figure 4.8-35. ECS Topology Domains Diagram

4.8.1.16.2 Domain Name Server Context

Figure 4.8-36 is the Domain Name Server context diagram.

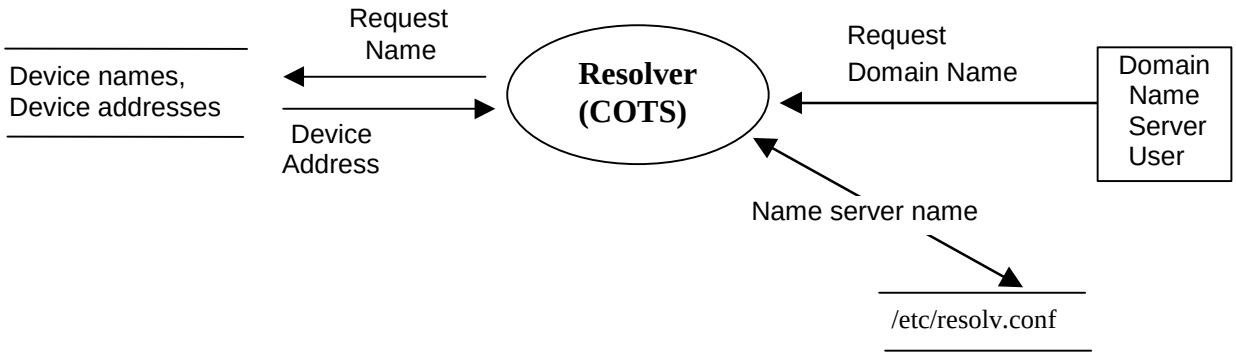


Figure 4.8-36. Domain Name Server Context Diagram

4.8.1.16.3 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.

4.8.1.16.4 Domain Name Server Process Descriptions

Table 4.8-52 provides descriptions of the Domain Name Server processes shown in the Domain Name Server context diagram.

Table 4.8-52. 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 /etc/resolv.conf is used as the place to start searching.

4.8.1.16.5 Domain Name Server Process Interface Descriptions

Table 4.8-53 provides descriptions of the interface events shown in the Domain Name Server architecture diagram.

Table 4.8-53. Domain Name Server Process Interface Events

Event	Event Frequency	Interface	Initiated by	Event Description
Request Domain Name	One per user request	*COTS Software:* resolver	User	A DNS **user** requests data.
Name server name	One per search directory change	Data Store	*COTS Software:* 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	*COTS Software:* resolver	*COTS Software:* name server	Returns the resolved address to the domain name requester via the Resolver.
Request Name	One per domain name request	*COTS Software:* name server	*COTS Software:* resolver	The resolver retrieves the domain name (device name and address) for the name server from an internal file used by the COTS software.

4.8.1.16.6 Domain Name Server Data Stores

Table 4.8-54 provides descriptions of the data store shown in the Domain Name Server architecture diagram.

Table 4.8-54. Domain Name Server Data Stores

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

4.8.1.17 Infrastructure Libraries Group Description

4.8.1.17.1 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 4.8-55 provides descriptions of the infrastructures in the ILG.

Table 4.8-55. Infrastructure Libraries (1 of 2)

Library	Description
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, such as mode management, error and event logging, life-cycle services, and the CCS Middleware Naming Service.
Server Request Framework (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 CCS Middleware call message passing and persistent storage as a CSS support capability with the described features available by subsystem request.
Universal References (UR)	A Universal Reference provider object from C++ objects generates UR 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.
Error/Event Logging	Event/Error 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 (requests for data or services) that occur during the ECS data processing and distributing.
Message Passing (MP)	Message Passing provides peer-to-peer asynchronous communications service, which notifies clients of specific event triggers. This service is provided upon subsystem request by the CSS. Message Passing is used in ECS for communications 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.
ServerUR	Provides unique identification (universal reference) for data and service objects in the ECS. 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 CCS Middleware-based communication.

Table 4.8-55. Infrastructure Libraries (2 of 2)

Library	Description
Fault Handling (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 CCS Naming Service, 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.
DBWrapper directory	The DBWrapper directory is the DBMS Interface Infrastructure Library used by ECS applications to connect to the Sybase ASEs. Sybase ASEs operate by ECS defined guidelines for mode management, thread safety, error handling, error recovery, security, configuration management, and performance of database connections.

4.8.1.17.2 Infrastructure Libraries Group Context

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

4.8.1.17.3 Infrastructure Libraries Group Architecture

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

4.8.1.17.4 Infrastructure Libraries Group Process Descriptions

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

4.8.1.17.5 Infrastructure Libraries Group Interface Descriptions

Table 4.8-56 provides descriptions of the interfaces the Infrastructure Libraries Group.

Table 4.8-56. Infrastructure Libraries Group Interfaces (1 of 5)

Library	Interface	Initiated by	Library Description
Process Framework (PF)	*Library:* EcPf *Classes:* EcPfManagedServer, EcPfClient	EcDsScienceDataServer, EcDsHdfEosServer, EcDsDistributionServer, EcDsStRequestManagerServer, EcDsStArchiveServer, EcDsStPullMonitorServer, EcDsStStagingDiskServer, EcDsSt8MMServer, EcDsStDTFServer, EcDsStFtpServer, EcDsStCacheManagerServer, EcInPolling, EcInReqMgr, EcInEmailGWServer, EcInGran, EcClWbJdt, EcClDtUserProfileGateway, EcDmDictServer, EcDmDdMaintenanceTool, EcDmV0ToEcsGateway, EcPlSubMgr, EcPlOdMgr, EcDpPrDeletion, EcDpPrJobMgmt, EcDpPrJobMgmtClient, EcSbSubServer, EcGwDARServer, EcCsEmailParser, EcCsMojoGateway, EcCsRegistry, EcCsMtMGateway, EcMsAcRegUserSrvr, EcMsAcOrderSrvr	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. Features and services include: • Mode management • Error and event logging • Life-cycle services • CCS Naming Service

Table 4.8-56. Infrastructure Libraries Group Interfaces (2 of 5)

Library	Interface	Initiated by	Library Description
Server Request Framework (SRF)	*Library (Common):* srf *Classes:* EcSrRequestServer_C, EcSrAsynchRequest_C	EcInReqMgr, EcDmDictServer, EcPlSubMgr, EcSbGui, EcPlPREditor, EcDpPrDeletion, EcSbSubServer, EcGwDARServer, EcCsMojoGateway, EcMsAcRegUserSrvr, EcDsScienceDataServer	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 CCS Middleware calls, message passing and persistent storage as a CSS support capability with the described features available by subsystem request.
Universal References (UR)	*Library (Common)*: EcUr	Object Origination	A Universal Reference provider object from C++ objects generates UR 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.

Table 4.8-56. Infrastructure Libraries Group Interfaces (3 of 5)

Library	Interface	Initiated by	Library Description
Error/Event Logging	*Library:* event *Class:* EcLgErrorMsg	EcInReqMgr, EcInGran, EcInPolling, EcInGUI, EcInEmailGWServer, EcDmDictServer, EcPlSubMgr, EcPlOdMgr, EcDpPrDeletion, EcDpPrJobMgmt, EcGwDARServer, EcCsEmailParser, EcCsMojoGateway, EcMsAcRegUserSrvr, EcMsAcOrderSrvr, EcDsScienceDataServer, EcDsHdfEosServer, EcDsDistributionServer, EcDsStRequestManagerServer, EcDsStArchiveServer, EcDsStCacheManagerServer, EcDsStStagingDiskServer, EcDsSt8MMServer, EcDsStDTFServer, EcDsStFtpServer, EcDmV0ToEcsGateway, EcSbSubServer, and EcCsMtMGateway	Event/Error 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 (requests for data or services) that occur during the ECS data processing and distributing.
Message Passing (MP)	*Library:* EcDcMsgPsng1	*Processes:* EcSbSubServer, EcPlSubMgr	Message Passing provides peer-to-peer asynchronous communications service, which notifies clients of specific event triggers. This service is provided upon subsystem request by the CSS. Message Passing is used in ECS for communications 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.

Table 4.8-56. Infrastructure Libraries Group Interfaces (4 of 5)

Library	Interface	Initiated by	Library Description
ServerUR	*Library (Common):* EcUr *Class:* EcUrServerUR	*Processes:* EcDmDdMaintenanceTool, EcSbGui *Classes:* EcNsServiceLoc, EcCsMojoGateway, EcClSubscription, EcSbSubscriptionRServer, *DSS Libraries:* DsBt, DsDe1, DsGe	Provides unique identification (universal reference) for data and service objects in the ECS. 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 CCS Middleware-based communication.
Fault Handling (FH)	*Library:* EcFh *Class:* EcFhExecutor	EcDmDictServer, EcDpPrDeletion, EcCsEmailParser, EcGwDARServer, EcCsMojoGateway, EcMsAcRegUserSrvr, EcMsAcOrderSrvr, EcDsScienceDataServer, EcDsHdfEosServer, EcDsDistributionServer, EcDsStRequestManagerServer, EcDsStArchiveServer, EcDsStCacheManagerServer, EcDsStStagingDiskServer, EcDsSt8MMServer, EcDsStDTFServer, EcDsStFtpServer, EcSbSubServer	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 CCS Naming Service, 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.
DBWrapper directory	*Libraries:* EcPoDbRW, EcPoDb *Class:* EcPoConnectionsRW	*Processes:* EcDsStArchiveServer, EcDsStStagingDiskServer, EcDsSt8MMServer, EcDsStDTFServer, EcDsStmgtGui, EcDmDictServer, EcSbSubServer	This is the DBMS Interface Infrastructure Library. Sybase ASEs implement ECS defined guidelines for mode management, thread safety, error handling, error recovery, security, configuration, and performance of database connections.

Table 4.8-56. Infrastructure Libraries Group Interfaces (5 of 5)

Library	Interface	Initiated by	Library Description
Time Service	*Libraries:* EcTiTimeService, *Class:* EcTiTimeService		This class provides the structured current time information and RogueWave time information.

4.8.1.17.6 Infrastructure Library Group Data Stores

Data Stores are not applicable for the Infrastructure Library Group.

4.8.2 The Distributed Computing Configuration Item Context

Figure 4.8-37 is the Distributed Computing Configuration Item (DCCI) CSCI context diagrams. The diagrams show the events sent to the DCCI CSCI and the events the DCCI CSCI sends to other CSCIs. Table 4.8-57 provides descriptions of the interface events shown in the DCCI CSCI context diagrams.

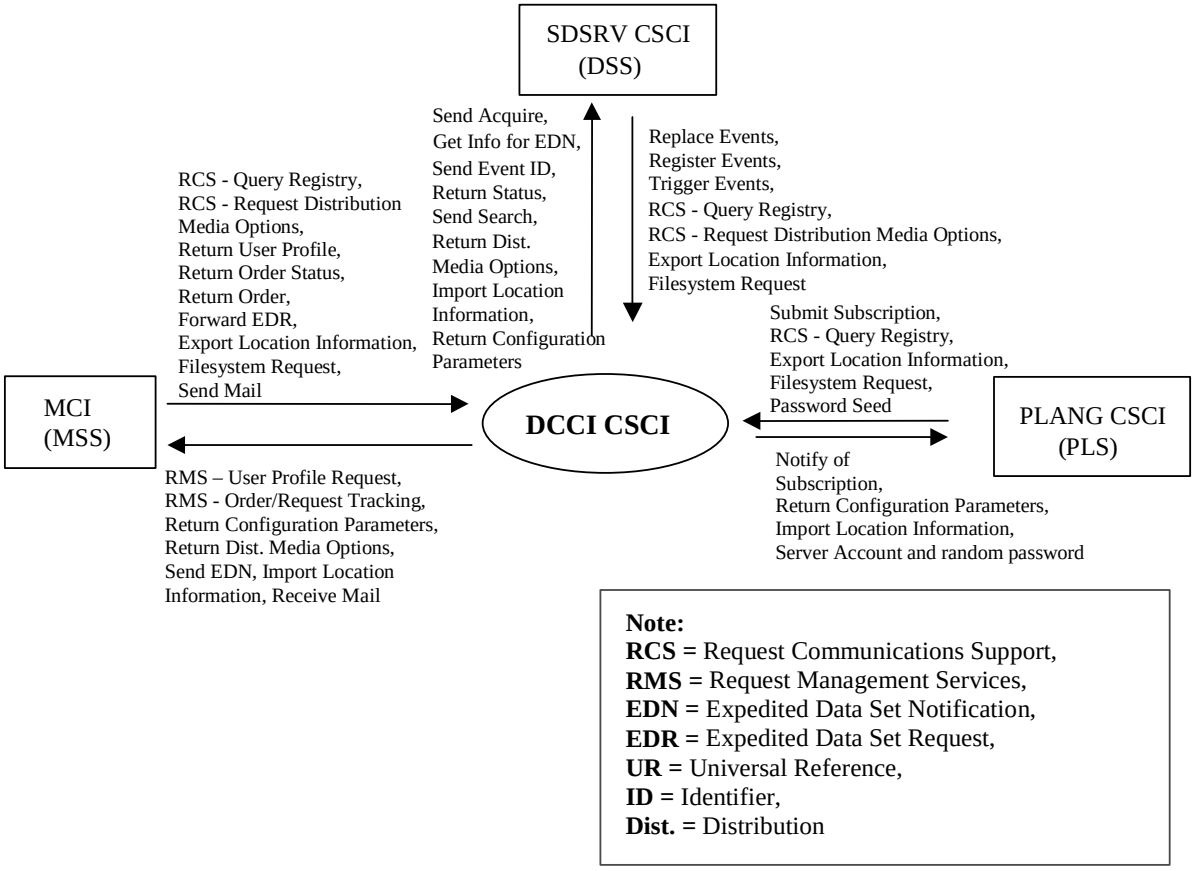


Figure 4.8-37. Distributed Computing Configuration Item (DCCI) CSCI Context Diagram

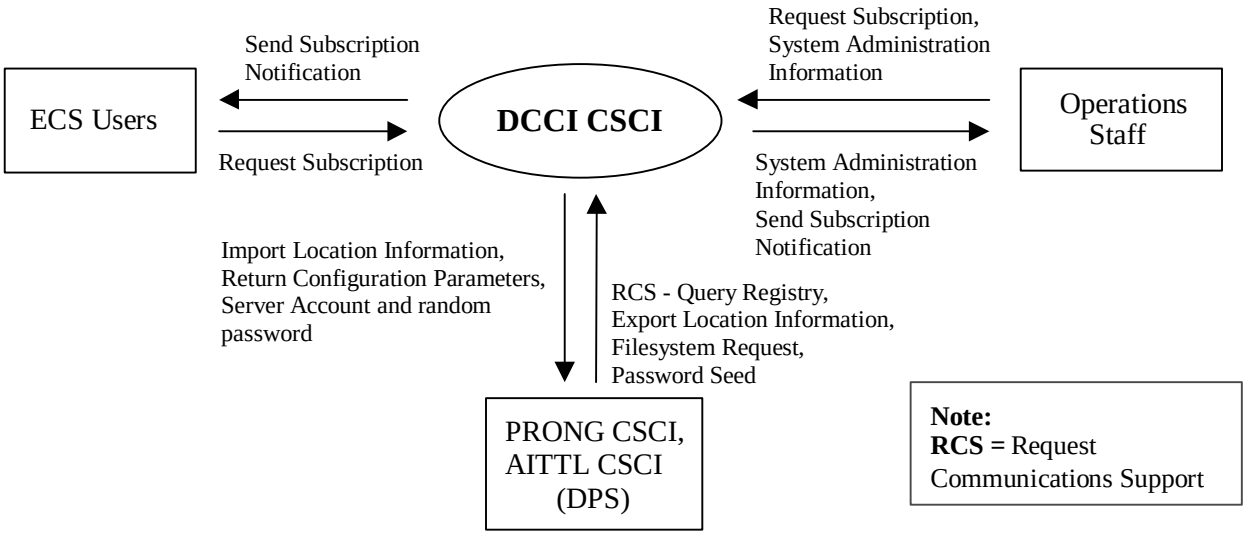


Figure 4.8-37. Distributed Computing Configuration Item (DCCI) CSCI Context Diagram (cont.)

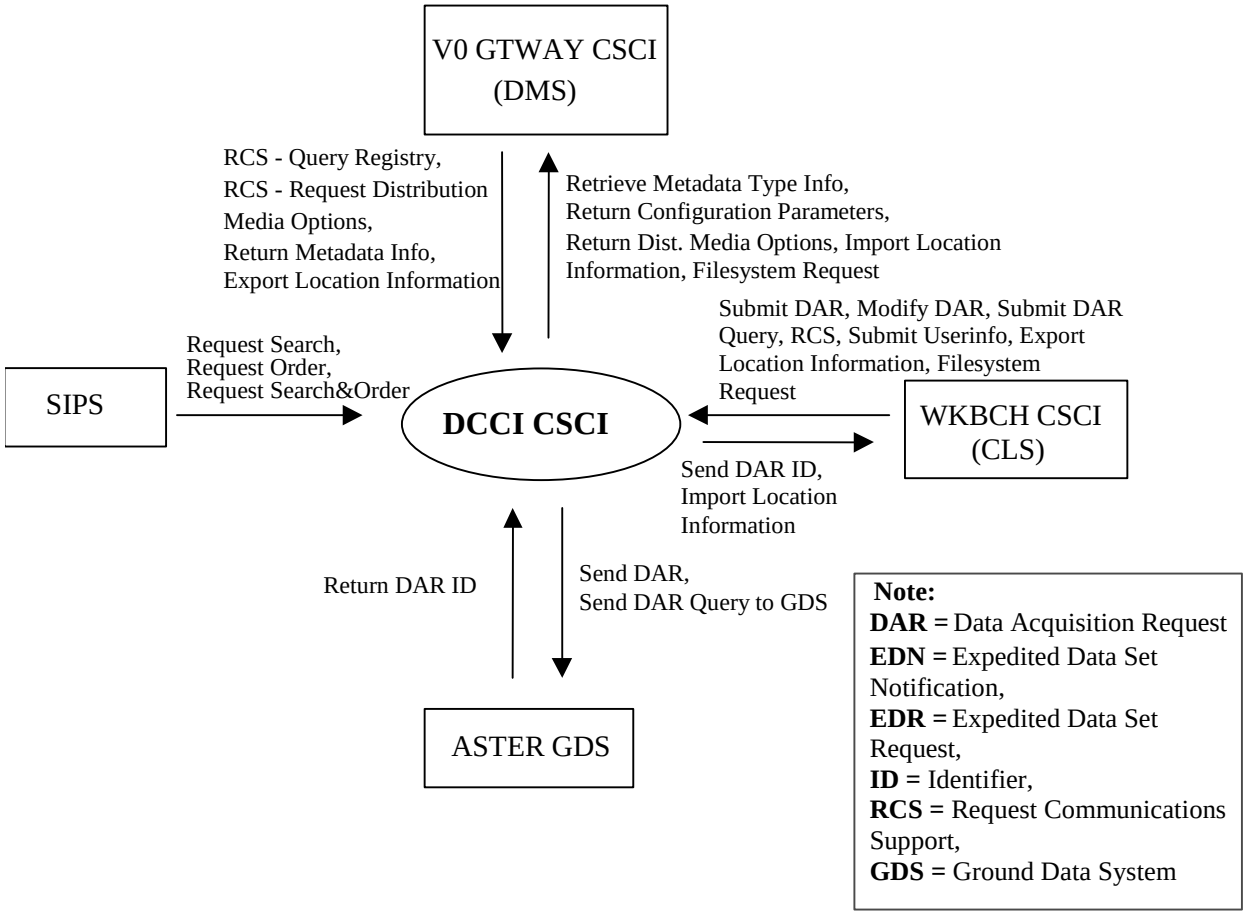


Figure 4.8-37. Distributed Computing Configuration Item (DCCI) CSCI Context Diagram (cont.)

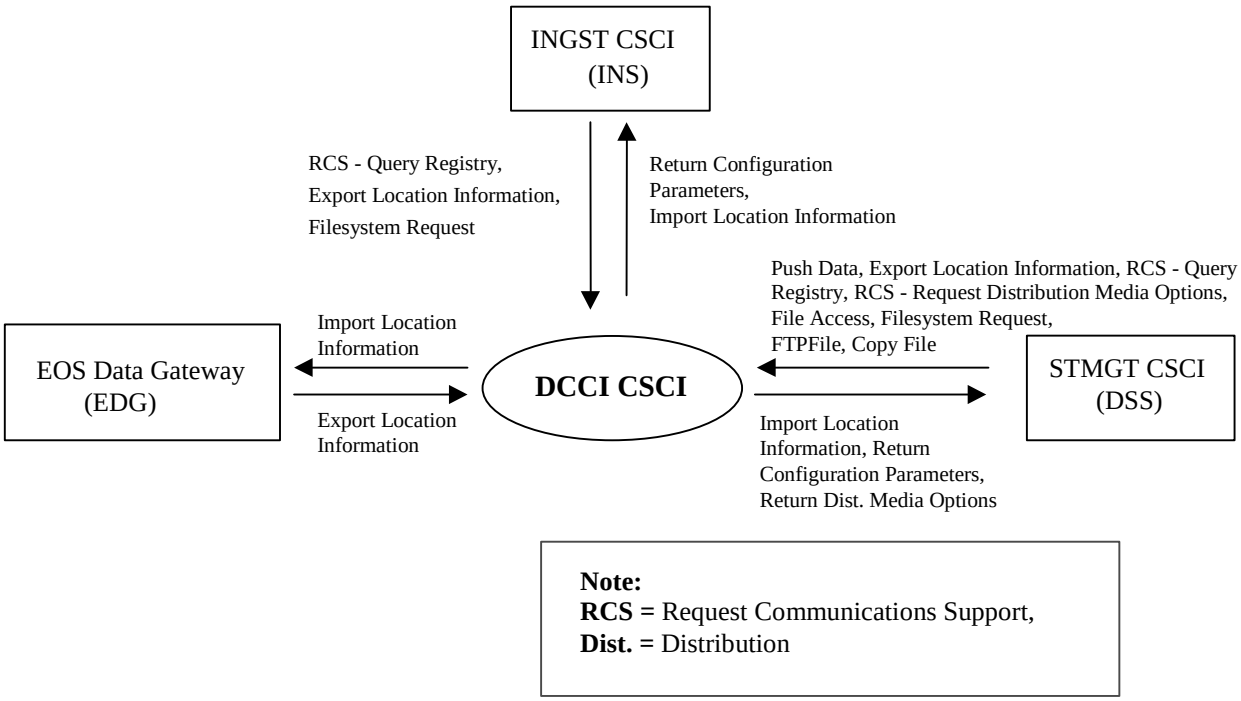


Figure 4.8-37. Distributed Computing Configuration Item (DCCI) CSCI Context Diagram (cont.)

Table 4.8-57. Distributed Computing Configuration Item (DCCI) CSCI Interface Events (1 of 7)

Event	Interface Event Description
Send Acquire	An “acquire” (instruction to obtain data) is created by the DCCI CSCI and sent to the **SDSRV CSCI** via CCS Middleware calls. This is similar to the “Request Product” interface event, except it applies to EDOS expedited data.
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
Send Search	The DCCI CSCI (Machine-to-Machine Gateway Server CSC) sends search requests received via the SIPS interface to the **SDSRV CSCI** on behalf of an external ECS user.
Return Dist. Media Options	The Configuration Registry CSC returns the requested distribution media options to the **SDSRV** and **MCI CSCIs.**
Import Location Information	The **SDSRV, PLANG and MCI CSCIs** and the **SBSRV CSC** request server location information from the CCS Name Server.
Return Configuration Parameters	The DCCI CSCI returns the requested configuration parameters to the **SDSRV, PLANG, MCI CSCIs and the SBSRV, ASTER DAR Gateway and MTMGW CSCs**.
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 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.

Table 4.8-57. Distributed Computing Configuration Item (DCCI) CSCI Interface Events (2 of 7)

Event	Interface Event Description
Request Communications Support	The DCCI CSCI provides a library of services available to the **SDSRV, PLANG** and **MCI CSCIs**. The CSCI services required to perform specific assignments are requested from the DCCI CSCI. These services include: • CCS Middleware Support • Database Connection Services • 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 • Mode Information • Query Registry - Retrieving the requested configuration attribute-value pairs from the Configuration Registry • Request Distribution Media Options from the Configuration Registry
Export Location Information	The DCCI CSCI stores physical and logical location information received from the **SDSRV, PLANG** and **MCI CSCIs** in the CCS Name Server.
Filesystem Request	The NFS clients (via **SDSRV, PLANG** and **MCI CSCIs)** request ECS 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.
Submit Subscription	The **PLANG CSCI** submits a subscription request to the DCCI CSCI (SBSRV CSC) using the advertisement subscribing to an insert event for an ESDT.
Password Seed	The **PLANG CSCI** requests an account and provides a password seed to the CMI.
Notify of Subscription	The DCCI CSCI sends notification (via message passing) to the **PLANG CSCI** when the subscribed event occurs.
Server Account and random password	An account with random passwords, created for the server, is passed back to the server in the **PLANG** CSCI from the DCCI CSCI.

Table 4.8-57. Distributed Computing Configuration Item (DCCI) CSCI Interface Events (3 of 7)

Event	Interface Event Description
Request Management Services	The **MCI** 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 includes: • **System startup and shutdown -** Please refer to the release-related, current version of the Mission Operations Procedures for the ECS Project document (611) and the current ECS Project Training Material document (625), identified in Section 2.2.1 of this document. • **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. • **Order/Request Tracking** – The **MCI** provides an order tracking service to requesting subsystems to create and track user product orders by request.
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 **MCI** to forward to the ASTER GDS.
Receive Mail	The **MCI** requests electronic mail sent by ASTER GDS users to the DCCI CSCI mail server.
Return User Profile	The **MCI** returns user profiles to the DCCI CSCI (Subscription Server, ASTER DAR Gateway Server, MOJO Gateway Server and Machine-to-Machine Gateway Server CSCs) to authenticate users for use of ECS data and services.
Return Order Status	The **MCI** provides order ids and order status information for products requested by users.
Return Order	The MCI returns the product order object to the requester via the DCCI CSCI.
Forward 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).
Send Mail	The **MCI** sends electronic mail to the DCCI CSCI mail server to be stored for other users.
Return Configuration Parameters	The DCCI CSCI returns the requested configuration parameters to the **PRONG,** and **AITTL CSCIs**.
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.
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.
Send Subscription Notification	An **ECS user** or **Operations Staff** member receives notification the subscription event has occurred.

Table 4.8-57. Distributed Computing Configuration Item (DCCI) CSCI Interface Events (4 of 7)

Event	Interface Event Description
Request Communications Support	The DCCI CSCI provides a library of services available to the **PRONG and AITTL CSCIs**. The CSCI services required to perform specific assignments are requested from the DCCI CSCI. These services include: • CCS Middleware Support • Database Connection Services • 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 • Mode Information • Query Registry - Retrieving the requested configuration attribute-value pairs from the Configuration Registry • Request Distribution Media Options from the Configuration Registry
Export Location Information	The DCCI CSCI stores physical and logical location information, received from the **PRONG** and **AITLL CSCIs,** in the CCS Name Server.
Filesystem Request	The NFS clients (via **PRONG and AITTL CSCIs)** request ECS 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.
Password Seed	The **AITTL** and **PRONG CSCIs** request an account and provide a password seed to the CMI.
Import Location Information	The **PRONG** and **AITTL CSCIs** request server location information from the CCS Name Server.
Server Account and random password	An account with random passwords, created for the server, is passed back to the server in the **PRONG** and **AITTL CSCIs** from the DCCI CSCI.
Retrieve Metadata Type Info	The DCCI CSCI (Machine-to-Machine Gateway Server CSC) retrieves type information for qualifying metadata specified in a SIPS search request from the **DDICT CSCI**.
Return Configuration Parameters	The DCCI CSCI returns the requested configuration parameters to the **DDICT and V0 GTWAY CSCIs**.
Return Dist. Media Options	The Configuration Registry CSC returns the requested distribution media options to the **DDICT and V0 GTWAY CSCIs**.
Import Location Information	The **DDICT, V0 GTWAY** and **WKBCH CSCIs** request server location information from the CCS Name Server.

Table 4.8-57. Distributed Computing Configuration Item (DCCI) CSCI Interface Events (5 of 7)

Event	Interface Event Description
Filesystem Request	The NFS clients (via **DDICT, V0 GTWAY** and **WKBCH CSCIs)** request ECS 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.
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 Communications Support	The DCCI CSCI provides a library of services available to the **WKBCH CSCI**. The CSCI services required to perform specific assignments are requested from the DCCI CSCI. These services include: • CCS Middleware Support • Database Connection Services • 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 • Mode Information • Query Registry - Retrieving the requested configuration attribute-value pairs from the Configuration Registry • Request Distribution Media Options from the Configuration Registry
Submit Userinfo	The **WKBCH CSCI** sends the user name and password to the DCCI CSCI. The DCCI CSCI returns a session id.
Export Location Information	The DCCI CSCI stores physical and logical location information, received from the **DDICT, V0 GTWAY** and **WKBCH CSCIs,** in the CCS Name Server.
Send DAR ID	The ASTER DAR Gateway Server extracts the returned DAR ID and sends it to the **WKBCH CSCI** 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.

Table 4.8-57. Distributed Computing Configuration Item (DCCI) CSCI Interface Events (6 of 7)

Event	Interface Event Description
Send DAR Query to GDS	The DCCI CSCI sends the DAR query to the **ASTER GDS** DAR Server.
Return DAR ID	The **ASTER GDS** returns a DAR ID to the ASTER DAR Gateway Server at the ECS.
Request Search	Search requests are sent to the DCCI CSCI (Machine-to-Machine Gateway Server CSC) via the **SIPS** interface.
Request Order	Order requests are sent to the DCCI CSCI (Machine-to-Machine Gateway Server CSC) via the **SIPS** interface.
Request Search&Order	Integrated search and order requests are sent to the DCCI CSCI (Machine-to-Machine Gateway Server CSC) via the **SIPS** interface.
Return Metadata Info	The DCCI CSCI receives metadata type information from the **DDICT CSCI**.
Return Configuration Parameters	The DCCI CSCI returns the requested configuration parameters to the **INGST** and **STMGT CSCIs**.
Import Location Information	The **EOS Data Gateway (EDG)** and **INGST** and **STMGT CSCIs** request server location information from the CCS Name Server.
Push Data	The **STMGT CSCI** pushes data (i.e., EDS), using 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.
Export Location Information	The DCCI CSCI stores physical and logical location information, received from the **EOS Data Gateway (EDG)** and **INGST** and **STMGT CSCIs,** in the CCS Name Server.
Request Communications Support	The DCCI CSCI provides a library of services available to the **STMGT and INGST CSCIs**. The CSCI services required to perform specific assignments are requested from the DCCI CSCI. These services include: • CCS Middleware Support • Database Connection Services • 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 • Mode Information • Query Registry - Retrieving the requested configuration attribute-value pairs from the Configuration Registry • Request Distribution Media Options from the Configuration Registry

Table 4.8-57. Distributed Computing Configuration Item (DCCI) CSCI Interface Events (7 of 7)

Event	Interface Event Description
File Access	The **STMGT CSCI** uses the BDS to transfer large data files over the high-speed Gig Ethernet interface for storage in the Science Data Server archives.
Filesystem Request	The NFS clients (via **DDICT, V0 GTWAY** and **WKBCH CSCIs)** request ECS 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.
FTPFile	The **STMGT CSCI** sends requests to the FTP Daemon to transfer the files to the Pull cache or to an external user.
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.
Return Dist. Media Options	The DCCI CSCI (Configuration Registry CSC) returns the requested distribution media options to the **STMGT** and **DDIST CSCIs**.

4.8.3 Distributed Computing Configuration Item Architecture

An architecture diagram is not applicable for the DCCI CSCI. However, Table 4.8-58 shows the mapping between SDPS/CSMS CSCIs and CSS CSCs.

Table 4.8-58. SDPS/CSMS CSCI to CSS CSC Mappings (1 of 3)

SDPS/CSMS CSCI	CSS CSC	Process Used	CSS Libraries Used
SDSRV	– CCS Middleware – Subscription Server (SBSRV) – E-mail Parser Gateway Server – Configuration Registry Server	– EcCsIdNameServer – EcSbSubServer – EcCsEmailParser – EcCsRegistry	– Process Framework (PF) – ServerUR, EcSbCl – Error Logging – Event Logging – Universal Reference (UR) – EcCsRegistry – CCS Middleware
STMGT	– CCS Middleware – File Transfer Protocol (FTP) – Network File System (NFS) – Filecopy – Configuration Registry Server	– EcCsIdNameServer – ftp_popen – NFS Client – EcUtFileCopy – EcCsRegistry	– PF – ServerUR – Error Logging – Event Logging – UR – EcCsRegistry – CCS Middleware

Table 4.8-58. SDPS/CSMS CSCI to CSS CSC Mappings (2 of 3)

SDPS/CSMS CSCI	CSS CSC	Process Used	CSS Libraries Used
INGST	– CCS Middleware – E-Mail Parser Gateway Server – FTP – NFS – Configuration Registry Server	– EcCsIdNameServer – EcCsEmailParser – ftp_popen – NFS Client – EcCsRegistry	– PF – ServerUR – Error Logging – Event Logging – UR – Fault Handling Services – Server Request Framework (SRF) – CCS Middleware
EOS Data Gateway	– CCS Middleware	– EcCsIdNameServer	– PF – ServerUR – UR – Error Logging – Event Logging – CCS Middleware
WKBCH	– CCS Middleware – MOJO Gateway Server	– EcCsIdNameServer – EcCsMojoGateway	– PF – ServerUR – Error Logging – Event Logging – UR – CCS Middleware
ODFRM	– Configuration Registry Server	– EcCsRegistry	– PF – ServerUR – Error Logging – Event Logging – UR – CCS Middleware
Desktop	N/A	N/A	– PF – ServerUR – Error Logging – Event Logging – UR – CCS Middleware
DDICT	– CCS Middleware – Configuration Registry Server	– EcCsIdNameServer – EcCsRegistry	– PF – ServerUR – Error Logging – Event Logging – UR – CCS Middleware

Table 4.8-58. SDPS/CSMS CSCI to CSS CSC Mappings (3 of 3)

SDPS/CSMS CSCI	CSS CSC	Process Used	CSS Libraries Used
V0 Gateway	– CCS Middleware – Configuration Registry Server	– EcCsIdNameServer – EcCsRegistry	– PF – ServerUR – Error Logging – Event Logging – UR – CCS Middleware
PLANG	– CCS Middleware – SBSRV – Configuration Registry Server – Cryptographic Management Interface	– EcCsIdNameServer – EcSbSubServer – EcCsRegistry – CMI	– PF – ServerUR – Message Passing – Error Logging – Event Logging – UR – CCS Middleware
PRONG	– CCS Middleware – Configuration Registry Server – Cryptographic Management Interface	– EcCsIdNameServer – EcCsRegistry – CMI	– PF – ServerUR – Error Logging – Event Logging – CCS Middleware
AITTL	– CCS Middleware – Configuration Registry Server – Cryptographic Management Interface	– EcCsIdNameServer – EcCsRegistry – CMI	– PF – ServerUR – Error Logging – Event Logging – CCS Middleware
MCI (CSMS)	– E-Mail Parser Gateway Server – SBSRV – ASTER DAR Gateway Server – MOJO Gateway Server – Configuration Registry Server	– EcCsEmailParser – EcSbSubServer – EcGwDARServer – EcCsMojoGateway – EcCsRegistry	– PF – ServerUR – Error Logging – Event Logging – UR – CCS Middleware

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.

4.8.7 Communications Subsystem Hardware CI Description

Document 920-TDx-001 provides descriptions of the Distributed Computing Configuraton HWCI.

Three DCCI software programs run on this host including the, Domain Name Server (DNS) and Mail Server. 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.

The Communications Subsystem DCCI uses other hosts and various hardware configuration items.

The Subscription Server (SBSRV), ASTER DAR Communications Gateway, and ASTER E-Mail Parser 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.

The SMC provides two Sun servers to receive data from external (non-EMSn) data providers. These serversare the International Ground Station (IGS) FTP Servers.

The Subscription Server (SBSRV), ASTER DAR Communications Gateway, and ASTER E-mail Parser Gateway are stored to local disk on the DMS Interface Server. The CSS Server is a stand-alone host and intrinsically does not have fail-over capability. DNS and Distributed Time Service (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.

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 Monitoring 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 diagrams. The external systems referred to in the context diagrams are EDOS, ASTER, NASA Internet Science Network (NISN), Version 0 (V0) Information Management System (IMS), ESDIS, Science Users, and the Landsat 7 Archive Management System (LAMS). Table 4.9-1 provides descriptions of the interface events shown in the MSS context diagrams.

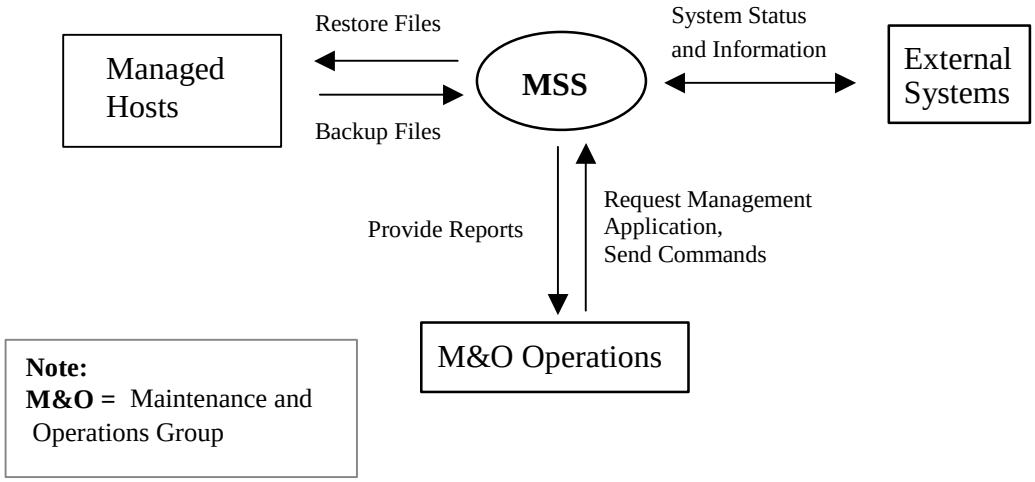


Figure 4.9-1. System Management Subsystem Context Diagram

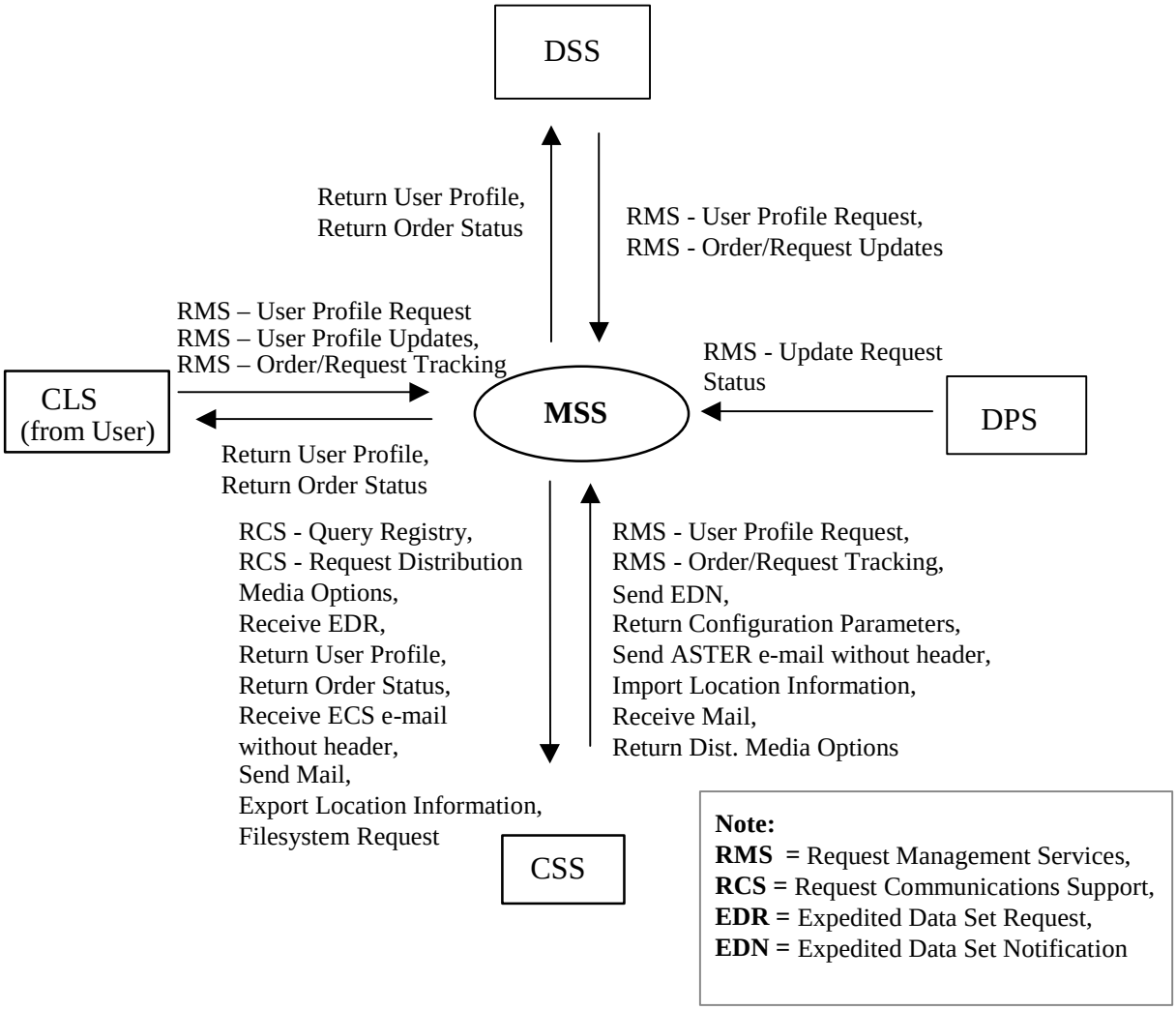
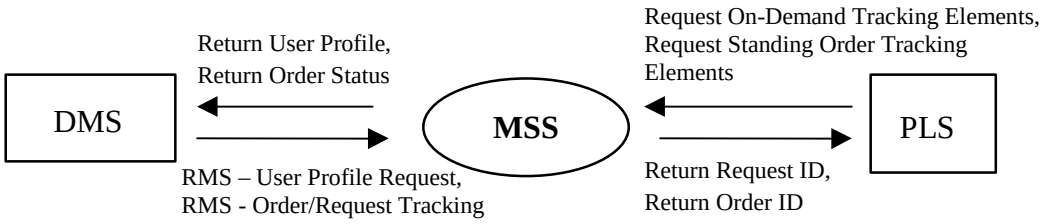


Figure 4.9-1. System Management Subsystem Context Diagram (cont.)



Note:
RMS = Request Management Services,
ID = Identifier

Figure 4.9-1. System Management Subsystem Context Diagram (cont.)

Table 4.9-1. System Management Subsystem Interface Events (1 of 4)

Event	Interface Event Description
System Status and Information	The MSS exchanges system status, trouble reports, and management report information with **external systems** such as the ASTER Gorund Data System (GDS), NASA Internet Science Network (NISN), V0 Information Management System (IMS), and the Landsat 7 Level 1 Product Generation System (LPGS) via the EOSDIS Mission Support Network (EMSn).
Request Management Application	The **Maintenance and Operations** (**M&O**) **staff** interacts 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.
Send Commands	The **M&O Staff** issue commands to the MSS (Security Service CSC utilities) to exercise system security setup.
Provide Reports	The MSS (Security Service CSC utilities) perform their functions and report results to the **M&O Staff**.
Backup Files	Data is passed from the **Managed Hosts** (the client) to the MSS (the server) and archived to tape.
Restore Files	Data is passed to **Managed Hosts** (the client) to restore lost data from the tape backups from the MSS (the server).

Table 4.9-1. System Management Subsystem Interface Events (2 of 4)

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 includes: • **System startup and shutdown** - Please refer to the release-related, current version of the Mission Operations Procedures for the ECS Project document (611) and the current ECS Project Training Material document (625), identified in Section 2.2.1 of this document. The MSS also interfaces with other subsystems to perform the following: • **User Profile Request** - The MSS provides requesting subsystems (**DSS, CSS, CLS and DMS**) with User Profile information such as e-mail address and shipping address to support their processing activities. • **Order/Request Tracking** - The **CLS** (ODFRM CSCI, via the iPlanet Web Server), 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. In addition, the **CSS** (MTMGW CSC) and **DMS** (V0 GTWAY) can also create a user product order and retrieve it from the MSS. • **User Profile Updates** – The MSS receives user profile parameter updates from a user (via **CLS**) and makes the updates in the user profile database. • **Order/Request Updates** - The **DSS** (DDIST CSCI) interfaces with the Accountability Management Service Order/Request Tracking to create/update the user product order request such as media id, quantity and type. • **Update Request Status** - The **DPS** requests 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).
Send EDN	The **CSS** sends the EDN to the MSS to forward to the ASTER GDS.
Return Configuration Parameters	The **CSS** returns the requested configuration parameters to the MSS.
Send ASTER e-mail without header	The **CSS** sends an e-mail message to a predefined ASTER e-mail alias within the ECS (in MSS), without a header (e.g., Expedited Data Set Notification or EDN).
Import Location Information	The **CSS** returns physical and logical server location information to the MSS.
Receive Mail	The **CSS** returns mail to the MSS to be distributed to addressees.
Return Dist. Media Options	The **CSS** returns distribution media options to the MSS.

Table 4.9-1. System Management Subsystem Interface Events (3 of 4)

Event	Interface Event Description
Request Communications Support	The **CSS** provides a library of services available to each SDPS and CSMS subsystems. The subsystem services required to perform specific assignments are requested from the CSS. These services include: • CCS Middleware Support • Database Connection Services • File Transfer Services • Network & Distributed File Services • Bulk Data Transfer Services • Name/Address Services • Password Services • Server Request Framework (SRF) • Universal Reference (UR) • Error/Event Logging • Fault Handling Services • Mode Information • Query Registry - Retrieving the requested configuration attribute-value pairs from the Configuration Registry • Request Distribution Media Options from the Configuration Registry
Receive EDR	The MSS strips the EDR header and sends the EDR to the **CSS** (E-mail Parser Gateway Server CSC).
Return User Profile	The MSS returns user profile information requested by users to the **DSS, CSS** and **CLS**.
Return Order Status	The MSS provides order ids and order status information to the **CLS, PLS, DPS,** and **CSS** for products requested by users.
Receive 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 by the **CSS**.
Send Mail	The MSS uses the CsEmMailRelA interface to send mail to the **CSS**.
Export Location Information	The MSS stores physical and logical server location information in the **CSS**.
Filesystem Request	The **CSS** (NFS client) receives requests for ECS files and directories via an established mount point from the MSS. The CSS (NFS Server) makes the storage device(s) and its data accessible for use by the clients.
Request On-Demand Tracking Elements	The **PLS** requests on-demand tracking elements (i.e., Order ID and Request ID) from the MSS.
Request Standing Order Tracking Elements	The **PLS** requests standing order tracking elements of ASTER high level products (i.e., Order ID and Request ID) from the MSS.
Return Request ID	The **PLS** receives the request identifier from the MSS for on-demand tracking elements requests.
Return Order ID	The **PLS** receives the order identifier from the MSS for standing order tracking elements requests.

Table 4.9-1. System Management Subsystem Interface Events (4 of 4)

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 includes: • **System startup and shutdown** - Please refer to the release-related, current version of the Mission Operations Procedures for the ECS Project document (611) and the current ECS Project Training Material document (625), identified in Section 2.2.1 of this document. • **User Profile Request** - The MSS provides the **DMS** with User Profile information such as e-mail address and shipping address to support their processing activities. • **Order/Request Tracking** - The **DMS** interfaces with the Order/Request Tracking service to create and track a user product order.
Return User Profile	The MSS returns user profile information requested by users to the **DMS**.
Return Order Status	The MSS provides order ids and order status information to the **DMS** for products requested by users.
Return Order	An internal description of the order requested by the user (an object).

System Management Subsystem Structure

The MSS has two CSCIs and one hardware CI. The two CSCIs are the Management Software CSCI (MCI) 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 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 and MLCI.

Use of COTS in the System 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.

Note: The RogueWave libraries mentioned below are currently delivered with ECS custom code as static libraries. A separate installation of the libraries is no longer required.

- RogueWave's Tools.h++

  The Tools.h++ class libraries provide strings and collections.

- 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.

- 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 Adaptive Server Enterprise (ASE)

  Sybase's ASE provides access for MSS to insert, update, and delete MSS database information. The Sybase ASE 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.

- 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.

- ClearCase Baseline Management (ClearCase BLM)

  ClearCase BLM custom tool used at Landover to manage the EMD Baseline Management System (EBIS). It provides both general and site specific baselines for COTS and Custom software, O/S patches for each EMD host, and UNIX kernel parameters for each EMD host.

- 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 used to support trouble ticketing and inventory, logistics, and maintenance (ILM) activities at the DAACs and the SMC. The Remedy Trouble Ticket application provides the capability to electronically compose, submit and update trouble tickets and report the status of trouble tickets. 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. The Remedy ILM application provides the capability to track EMD hardware inventory, logistics, software licenses, and maintenance transactions.

- Sun ONE Web Server, Enterprise Edition

  The Sun ONE Web Server provides world-wide web services for ECS applications such as ODFRM and Trouble Ticketing. For example, it 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.

- Tcl/Tk

  The Tool Command Language is a scripting langauge, which runs on multiple platforms. As an interpreted langauge each statement is read in, parsed and executed in runtime. Tcl provides most of the handy utility functions shell scripts do to go through directories and sort the file names, execute commands and so on. An associate add-on toolkit allows a user to quickly create graphical applications without delving into packages like Win32, Motif and the X toolkit. With a surprisingly small amount of code, a user can quickly develop graphical applications.

- CCS Middleware Client

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

Error Handling and processing

When an error occurs, the error is logged into the applications log (ALOG). The Communications Subsystem (CSS) and System Management Subsystem (MSS) have two main mechanisms to handle the error:

1. Return an error status
2. Throw an exception

The MSS uses the following classes for error handling and processing:

The EcUtStatus class is used to describe the operational status of many functions. The values most often reported are "failed" and "ok." But depending upon the application, detailed values could be set and sent. Please refer to the definition of this class (located in /ecs/formal/COMMON/CSCI_Util/src/Logging/EcUtStatus.h) for all possible values.

The EcPoError class defines the basic error types and handling functions for using the EcPoConnectionsRW class (based upon RWDBTool++). The Subscription Server and MSS Order tracking Server use the EcPoConnectionsRW class.

The MsAcDatabaseError class defines the error types and access Application Program Interfaces (APIs) for MSS database access.

The RWCString is used to store some status value returned by applications.

Integer is used to return some error status by applications. This is used specifically between client and server communications.

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.

4.9.1.1.1 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 has 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

4.9.1.1.1.1 Network and Enterprise Management Framework Functional Overview

The network and enterprise management framework is a capability to monitor network devices and services and notify Operations staff when problems are detected. It consists of the WhatsUp Gold network-monitoring product, a PC-based COTS application that has a graphical user interface. WhatsUp Gold can discover and map a site's network devices automatically - identifying TCP/IP, NetBIOS or IPX devices and services on the devices – and track their status, providing Fault Management and Performance Management services for ECS as follows:

- Fault Management – Through use of standard protocols such as TCP/IP, WhatsUp Gold continuously polls mapped devices and services on the devices to detect when they go down. It can notify Operations staff by e-mail, pager, popup window, or other configurable means when problems are detected.
- Performance Management – WhatsUp Gold logs network status changes and polling statistics for each device. Operations staff can view graphs that depict devices by best or worst performance and that summarize device and service availability and response times. They can also obtain reports that identify device and service up and down times and accumulated polling statistics by device.

4.9.1.1.1.2 Network and Enterprise Management Framework Context

Figure 4.9-2 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 Framework CSC sends to other CSCIs or CSCs. Table 4.9-2 provides descriptions of the interface events shown in the Network and Enterprise Management Framework context diagram.

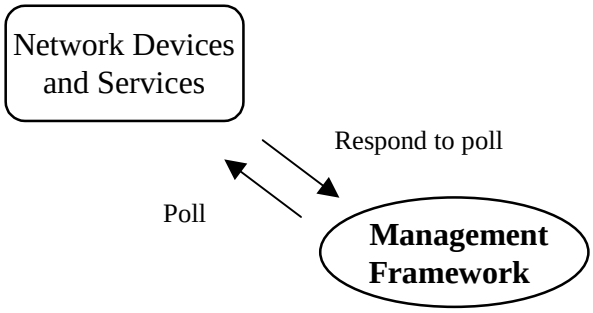


Figure 4.9-2. Network and Enterprise Management Framework Context Diagram

Table 4.9-2. Network and Enterprise Management Framework Interface Events

Event	Interface Event Description
Poll	Network and enterprise management framework polls devices and services on the devices to check on their status.
Respond to poll	Network devices and services acknowledge polling requests if they can.

4.9.1.1.1.3 Network and Enterprise Management Framework Process Architecture

Figure 4.9-3 is the Network and Enterprise Management Framework architecture diagram. The diagram shows the events sent to the Network and Enterprise Framework CSC processes and the events the Network and Enterprise Framework processes send to other processes.

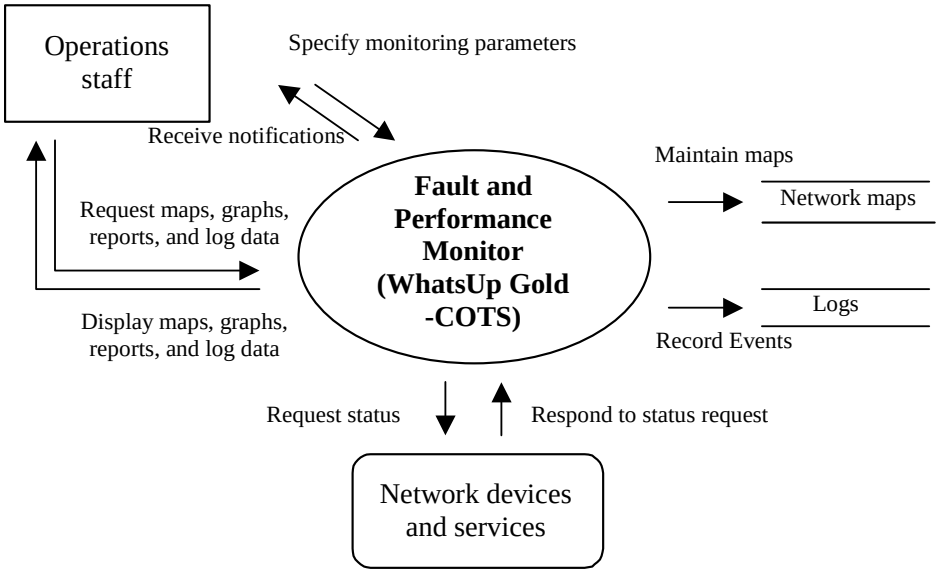


Figure 4.9-3. Network and Enterprise Management Framework Architecture Diagram

4.9.1.1.1.4 Network and Enterprise Management Framework Process Descriptions

Table 4.9-3 describes the processes in the Network and Enterprise Management Framework architecture diagram.

Table 4.9-3. Network and Enterprise Management Framework Processes

Process	Type	COTS/ Developed	Functionality
Fault and Performance Monitor	Server	COTS	The Fault and Performance Manager discovers network devices and services; determines their status, recording polling requests and responses as events; compiles event statistics, issues alarms when problems are detected; and produces graphs and reports in response to Operations staff requests.

4.9.1.1.1.5 Network and Enterprise Management Framework Process Interface Descriptions

Table 4.9-4 provides descriptions of the Network and Enterprise Management Framework interface events shown in the Network and Enterprise Management Framework architecture diagram.

Table 4.9-4. Network and Enterprise Management Framework Process Interface Events (1 of 5)

Event	Event Frequency	Interface	Initiated By	Event Description
Specify monitoring parameters	One per operator request	*Process:* Fault and Performance Monitor (COTS)	Operations staff	WhatsUp Gold permits operators to specify: • Devices and services to monitor • Polling methods and intervals • Subnets • Map colors and views • Alerts to be issued for each device and the triggering criteria for each • Notification types and content • Events to ignore
Request status	One per device per operator-specified time period	*Process:* Network devices and services	*Process:* Fault and Performance Monitor (COTS)	WhatsUp Gold polls a device by sending packets (echo requests) to the device using a monitoring method, polling frequency, time period, and timeout value specified by the operator. It monitors a service by communicating with the default port that the service runs on.

Table 4.9-4. Network and Enterprise Management Framework Process Interface Events (2 of 5)

Event	Event Frequency	Interface	Initiated By	Event Description
Respond to status requests	One per polling request	*Process:* Fault and Performance Monitor (COTS)	*Process:* Network devices and services	Network devices and services acknowledge a WhatsUp Gold status request.
Maintain maps	One per polling or update request	File	*Process:* Fault and Performance Monitor (COTS)	WhatsUp Gold creates and maintains network maps using auto discovery or parameters specified by the operator, and it uses polling results to continually update properties of the icons the maps contain.
Record events	One per polling request	Log	Fault and Performance Monitor (COTS)	WhatsUp Gold logs four types of data: • Syslog – logs standard UDP messages sent from routers, switches, UNIX hosts, etc. • Events – changes to network status, such as a device going down or coming back up • Polling statistics – accumulated round trip times of polls sent to a device. These statistics measure the availability and performance of a device • SNMP traps – displays all SNMP traps that have been received

Table 4.9-4. Network and Enterprise Management Framework Process Interface Events (3 of 5)

Event	Event Frequency	Interface	Initiated By	Event Description
Receive notifications	One per problem detected	Operations staff	*Process:* Fault and Performance Monitor (COTS)	If configured by an operator, Operations staff can receive one or more notifications when WhatsUp Gold detects an event. WhatsUp Gold can: • Sound an alarm • Activate a beeper • Execute a program • Send a message to a pager • Send a mail message • Send a pre-recorded messae to a telephone • Send a text to speech notification • Display a WinPopup • Send a group of notifications WhatsUp Gold can also send a network status report at a specified time interval via beeper, pager, or mail message as a recurring notification

Table 4.9-4. Network and Enterprise Management Framework Process Interface Events (4 of 5)

Event	Event Frequency	Interface	Initiated By	Event Description
Request graphs, reports, and log data	One per operator request	*Process:* Fault and Performance Monitor (COTS)	Operations staff	WhatsUp Gold permits operators to request: • Current status information about a device • Performance graphs by map, type (comprehensive, daily, monthly, day of the week, daily text/average, availability, hourly), device, graph format (bar or area chart), date range, data source, and output file type. Data in performance graphs can be sorted in ascending or descending device name order, and a report's view size can be adjusted • Event reports by map, type (summary, detail, or raw data) and date range. Data in summary and detail reports can be sorted in ascending, descending, or "worst first" order • Statistics reports by map, type (detail (by device), raw data, or day of the week) and date range. Data in detail reports can be sorted in ascending or descending device name order • Displays of WhatsUp Gold's logs

Table 4.9-4. Network and Enterprise Management Framework Process Interface Events (5 of 5)

Event	Event Frequency	Interface	Initiated By	Event Description
Display requested maps, graphs, reports, and logs	One per operator request	Operations staff	*Process:* Fault and Performance Monitor (COTS)	In response to operator requests, WhatsUp Gold provides operators with: • Maps of the devices in a network • Up-to-the-minute status of a device, including such information as its up/down state, number of times it was polled, round trip time of the last polling packet sent and received, poll history, and poll success rate • Performance graphs that show devices by best or worst performance based on aggregated polling statistics. The graphs can show summaries of device and service availability and response times • Event reports that show device up and down events, service up and down events and WhatsUp Gold events sucs as map open and close • Statistics reports that show round trip times and percentage of missed polls based on the accumulated polling statistics for each device • A GUI with which to browse WhatsUp Gold's log files

4.9.1.1.1.6 Network and Enterprise Management Framework Data Stores

Table 4.9-5 provides descriptions of the data stores used in the Network and Enterprise Management Framework architecture diagram.

Table 4.9-5. Network and Enterprise Management Framework Data Stores

Data Store	Type	Functionality
Syslog	File	Holds standard UDP messages sent from routers, switches, UNIX hosts, etc. Name is SL-*yyyy-mm-dd*.tab.
Event Log	File	Holds records that describe changes to network status, such as a device going down or coming back up. Name is EV-*yyyy-mm-dd*.tab.
Statistics Log	File	Holds polling statistic records - the accumulated round trip times of polls sent to a device - used to measure the availability and performance of a device. Name is ST-*yyyy-mm-dd*.tab.
SNMP Trap Log	File	Holds records of all SNMP traps that have been received. Name is SP-*yyyy-mm-dd*.tab.
Network Maps	File	Contains a network map's definition. Name is <*map_name*>.wup.

4.9.1.1.2 MCI - Security Service Computer Software Component Description

4.9.1.1.2.1 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 CCS Middleware, 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.

4.9.1.1.2.2 Security Service Context

Figure 4.9-4 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 4.9-6 provides descriptions of the interface events shown in the Security Service context diagram.

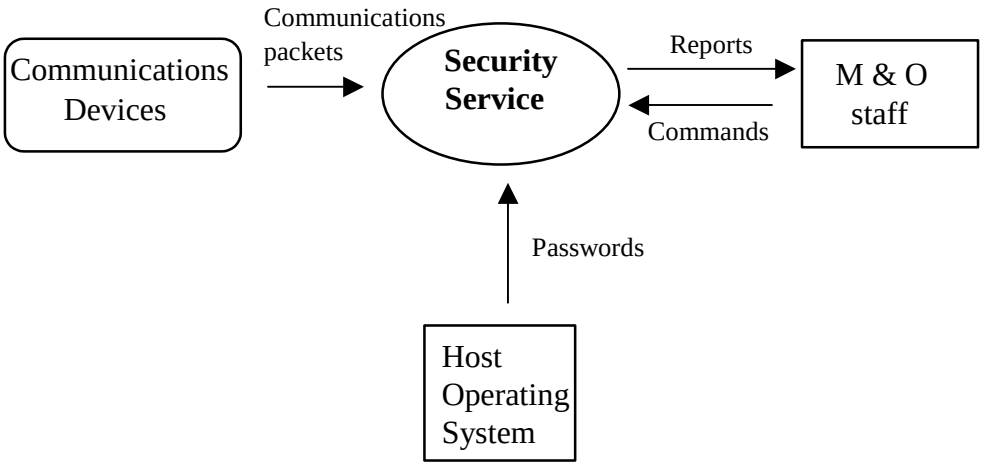


Figure 4.9-4. Security Service Context Diagram

Table 4.9-6. 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** issue 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 (in the **host operating system**) 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 (**communications devices**). The Security Service CSC analyzes packets for authorized sending sources.

4.9.1.1.2.3 Security Service Architecture

Figure 4.9-5 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.

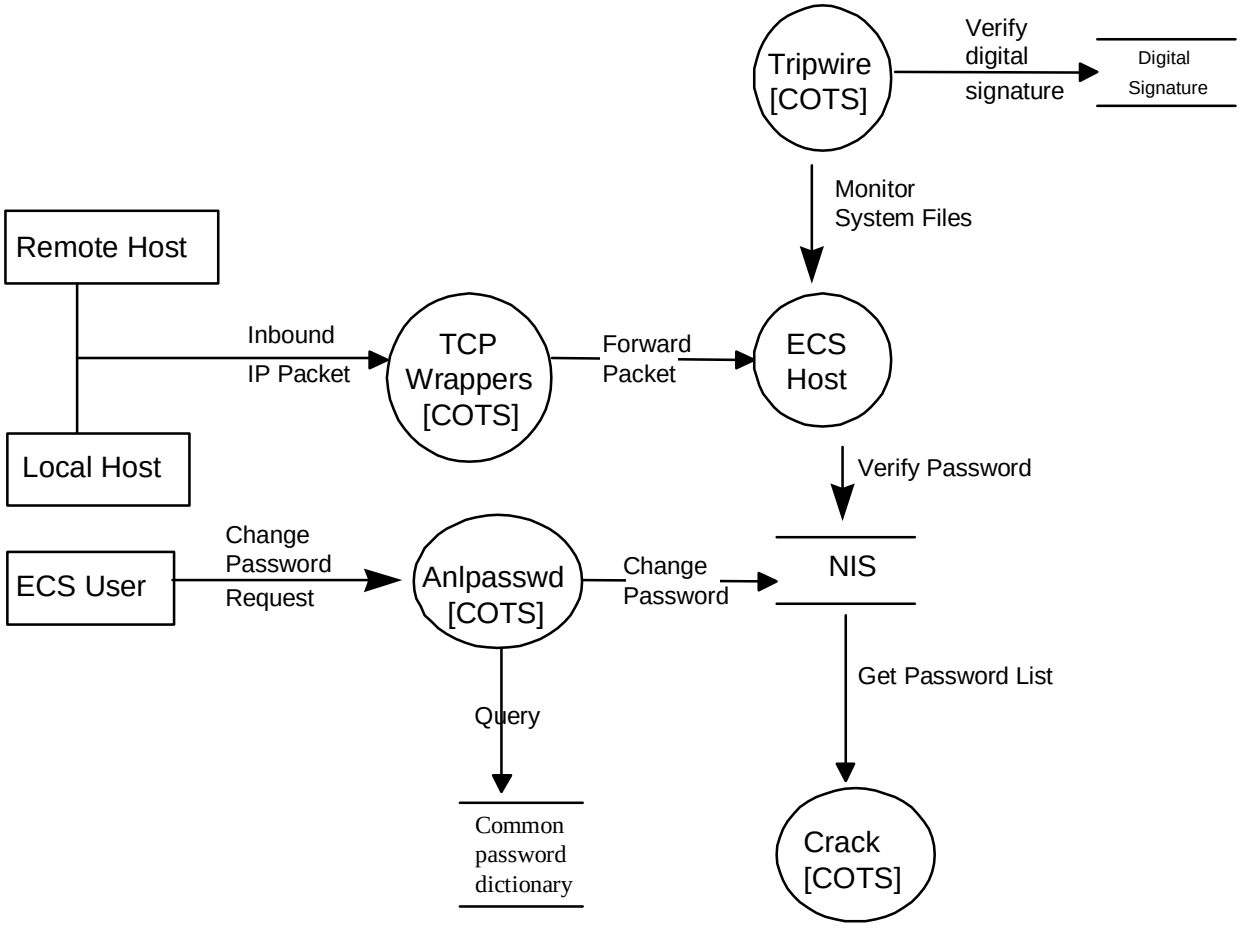


Figure 4.9-5. Security Service Architecture Diagram

4.9.1.1.2.4 Security Service Process Descriptions

Table 4.9-7 provides descriptions of the processes shown in the Security Service architecture diagram.

Table 4.9-7. Security Service Processes

Process	Type	COTS / Developed	Functionality
Anlpasswd	Other	COTS	Anlpasswd is a replacement for the standard UNIX passwd and yppasswd 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.
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.

4.9.1.1.2.5 Security Service Process Interface Descriptions

Table 4.9-8 provides descriptions of the interface events shown in the Security Service architecture diagram.

Table 4.9-8. Security Service Process Interface Events (1 of 2)

Event	Event Frequency	Interface	Initiated By	Event Description
Verify digital signature	Once per verify digital signature	*Data Store:* Digital Signature	*Process:* 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	*Process:* 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**.
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	*Data Store:* 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.

Table 4.9-8. Security Service Process Interface Events (2 of 2)

Event	Event Frequency	Interface	Initiated By	Event Description
Get password list	One per get password list	*Data Store:* NIS	*Process:* 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	*Data Store:* NIS	*Process:* 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	*Data Store:* Common Password Dictionary	*Process:* Anlpasswd (COTS)	The Anlpasswd makes a NIS system call to check the **common password dictionary** to ensure the attempted new password is not in this list.
Change password request	One per change password request	*Process:* Anlpasswd	User/Comm and line	An **ECS user** attempts to change their password and the Anlpasswd verifies the request that the new password does not contain any trivial or easy to guess password.
Inbound IP packet	One per inbound IP packet	*Process:* 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.

4.9.1.1.2.6 Security Service Data Stores

Table 4.9-9 provides descriptions of the data stores shown in the Security Service architecture diagram.

Table 4.9-9. 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. Anlpasswd verifies that the new password change does not include a word listed in the Anlpasswd file.
Digital Signature	Database	This proprietary database is used by Tripwire to record the digital signature for each system file it monitors.

4.9.1.1.3 MCI - Accountability Management Service Computer Software Component Description

4.9.1.1.3.1 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.

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 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.

4.9.1.1.3.2 Accountability Management Service Context

Figure 4.9-6 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 4.9-10 provides descriptions of the interface events shown in the Accountability Management context diagram.

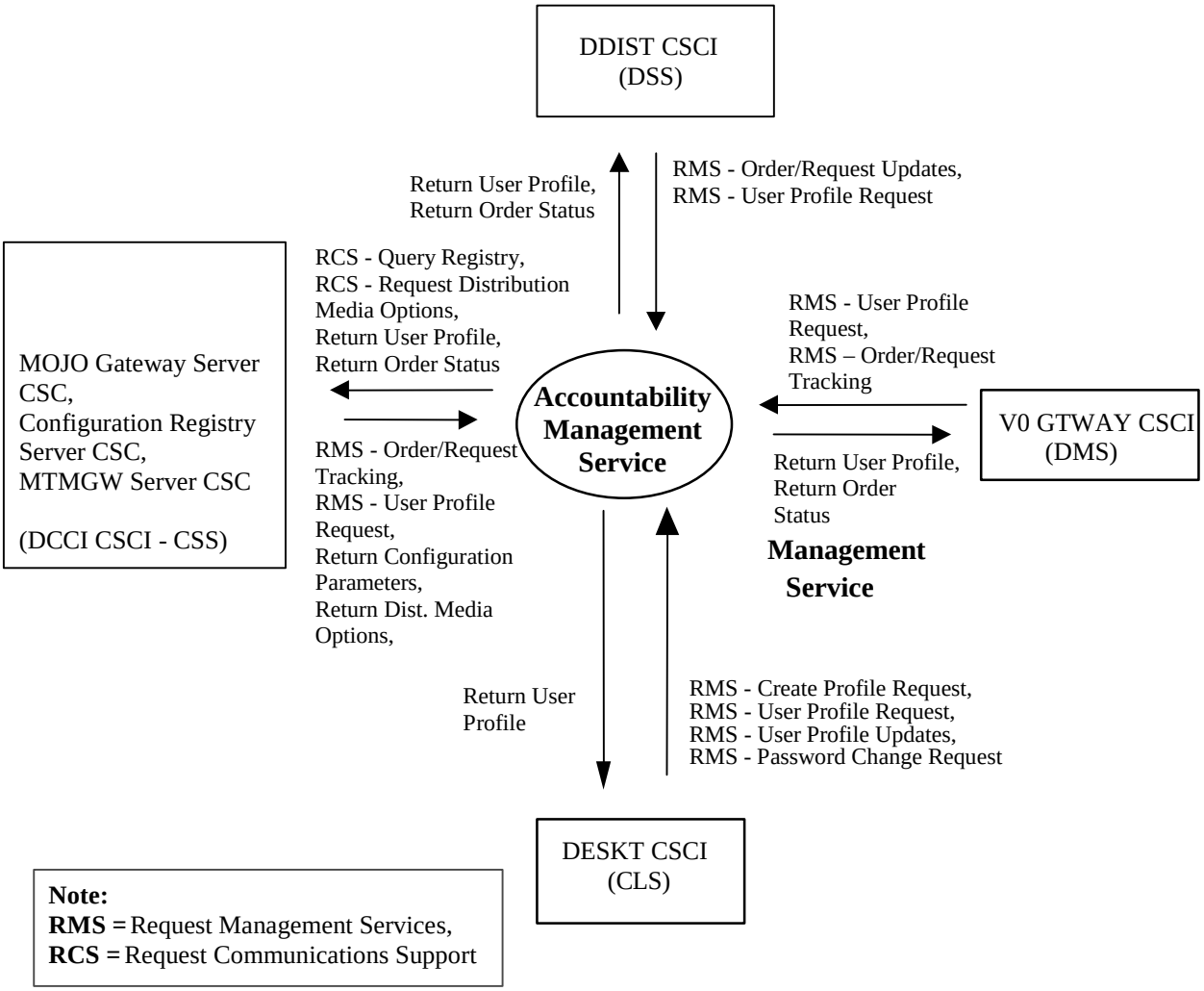


Figure 4.9-6. Accountability Management Service Context Diagram

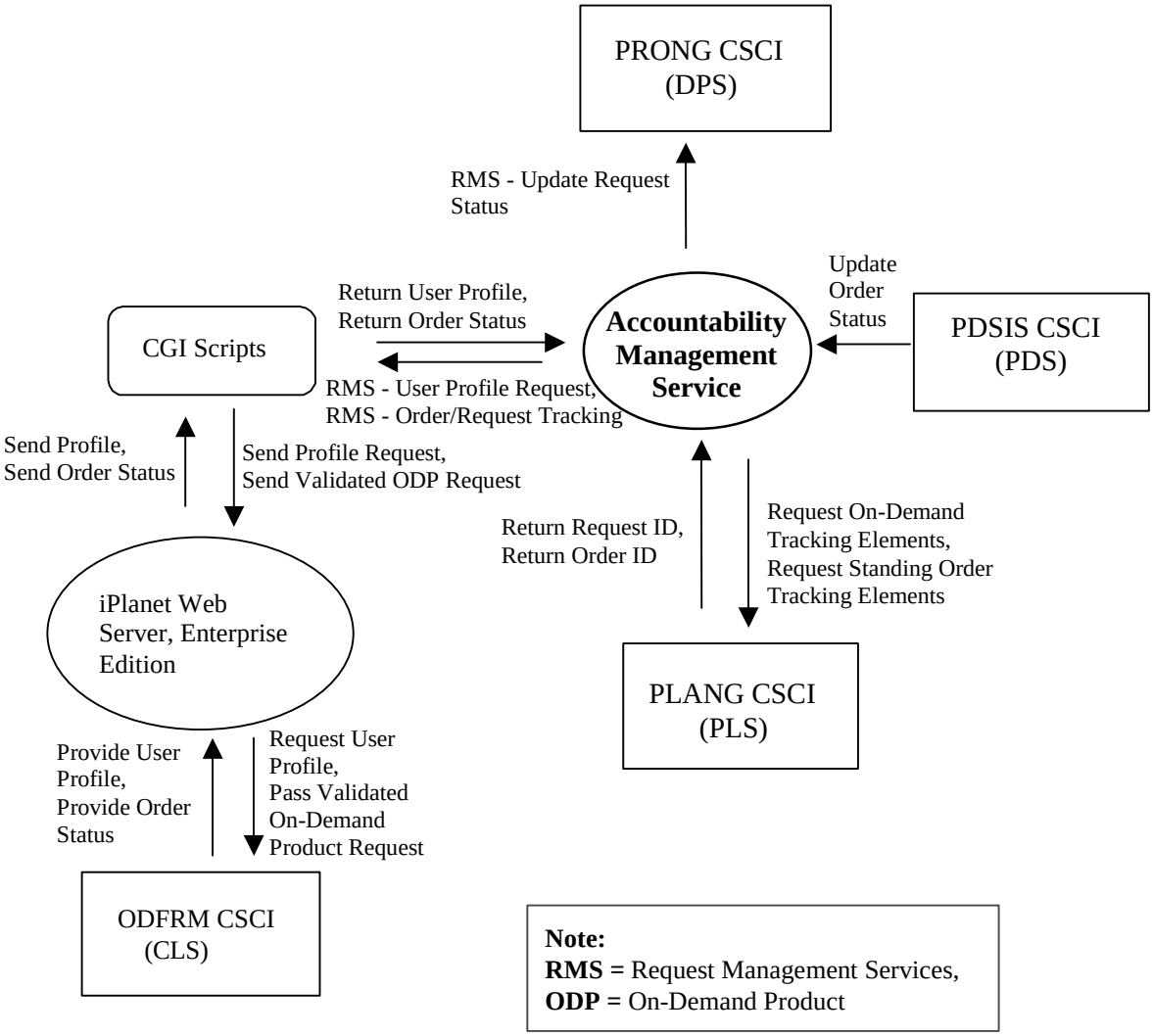


Figure 4.9-6. Accountability Management Service Context Diagram (cont.)

Table 4.9-10. Accountability Management Service Interface Events (1 of 3)

Event	Interface Event Description
Request Management Services	The MCI provides 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: The MCI interfaces with other CSCIs/CSCs to perform the following: • **Create Profile Request** - The MCI receives user information for becoming a registered user of the ECS from the DESKT CSCI. • **Order/Request Tracking** – The **MTMGW CSC** can create and track order or search and order requests received from the SIPS interface via the MCI Order Tracking Server. The **V0 GTWAY CSCI** interfaces with Accountability Management Service Order/Request Tracking service to create a user product order. • **Order/Request Updates** – The **DDIST CSCI** interfaces with the MCI Order Tracking Server to distribute and provide status on user data product orders. • **User Profile Request** – The Accountability Management Service provides the **DDIST**, **V0 GTWAY**, **DESKT** CSCIs and the **SBSRV**, **ASTER DAR Gateway Server, MTMGW Server** and **MOJO Gateway Server** CSCs with User Profile information 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 (via the **DESKT CSCI**) and makes the updates in the user profile database. This capability is available only at the SMC. • **Password Change Request** – The **DESKT CSCI** sends requests on behalf of ECS users to the MCI to change or reset users' authenticators in the MSS database.
Update Order Status	The Accountability Management Service CSC receives a request to update the order status from the **PDSIS CSCI**.
Return User Profile	The **DDIST CSCI**, **SBSRV CSC**, **ASTER DAR Gateway Server CSC**, **MOJO Gateway Server CSC, MTMGW Server CSC** and the **DESKT CSCI** receive user profile information from the Accountability Management Service to authenticate a user.
Return Order Status	The **DDIST CSCI**, **ASTER DAR Gateway Server CSC**, **MOJO Gateway Server CSC, MTMGW Server CSC, DESKT CSCI** and **V0 GTWAY CSCI** receive an order id and status for the requested ECS product from the Accountability Management Service.
Return Configuration Parameters	The **Configuration Registry CSC** returns the requested configuration parameters to the Accountability Management Service.
Return Dist. Media Options	The Accountability Management Service receives the distribution media options from the **Configuration Registry Server CSC**.

Table 4.9-10. Accountability Management Service Interface Events (2 of 3)

Event	Interface Event Description
Request Communications Support	The **DCCI CSCI** provides a library of services available to each SDPS and CSMS CSCI/CSC. The CSCI services required to perform specific assignments are requested from the DCCI CSCI. These services include: • CCS Middleware Support • Database Connection Services • Network & Distributed File Services • Name/Address Services • Password Services • Server Request Framework (SRF) • Universal Reference (UR) • Error/Event Logging • Fault Handling Services • Mode Information • Query Registry - Retrieving the requested configuration attribute-value pairs from the Configuration Registry • Request Distribution Media Options from the Configuration Registry
Request Management Services	The **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: The MCI interfaces with other CSCIs/CSCs to perform the following: • **Order/Request Tracking** - The **ODFRM CSCI** can create and track On-Demand or standing order requests sent through the iPlanet Web Server and CGI Interface to the MCI Order Tracking Server. • **User Profile Request** - The Accountability Management Service provides **the CGI Scripts** with User Profile information such as e-mail address and shipping address to support their processing activities. • **Update Request Status** - The **PRONG CSCI** requests the MCI 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).
Return User Profile	The Accountability Management Service CSC returns the requested user profile to a **CGI script**.
Return Order Status	The Accountability Management Service CSC returns the order status for the requested product to a **CGI script**.
Request On-Demand Tracking Elements	The **PLANG CSCI** requests on-demand, tracking elements (i.e., Order ID and Request ID) from the Accountability Management Service (MCI).
Request Standing Order Tracking Elements	The **PLANG CSCI** requests standing order tracking elements based upon orders of ASTER high-level products (i.e., Order ID and Request ID) from the Accountability Management Service (MCI).

Table 4.9-10. Accountability Management Service Interface Events (3 of 3)

Event	Interface Event Description
Return Request ID	The **PLANG CSCI** receives the request identifier from the Accountability Management Service for on-demand tracking elements.
Return Order ID	The **PLANG CSCI** receives the order identifier from the Accountability Management Service for standing order tracking elements.
Send Profile Request	The iPlanet Web Server sends the request information for a user profile to a CGI script.
Send Validated ODP Request	The iPlanet Web Server sends a request to a CGI script to access the MSS database. The EcClOdRequest process accesses the MSS database via CGI scripts and sends the user back the authentication.
Request User Profile	The ODFRM CSCI sends a request to the MSS Sybase ASE for a user to use ECS services or access ECS data products via the iPlaner Web Server using a CGI script.
Pass Validated On-Demand Product Request	The ODFRM CSCI sends a validated on-demand request to the iPlanet Web Server on behalf of a user for ECS products. The iPlanet Web Server spawns the process EcClOdRequest with the User Login information to pass on the validated on-demand product request.
Provide User Profile	The iPlanet Web Server returns the user profile to the ODFRM CSCI.
Provide Order Status	The iPlanet Web Server returns the status of an order for an ECS product(s) to the ODFRM CSCI.
Send Profile	The CGI scripts provide the user profile to the iPlanet Web Server.
Send Order Status	The CGI scripts provide an order id and status for the requested ECS product to the iPlanet Web Server.

4.9.1.1.3.3 Accountability Management Service Architecture

Figure 4.9-7 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.

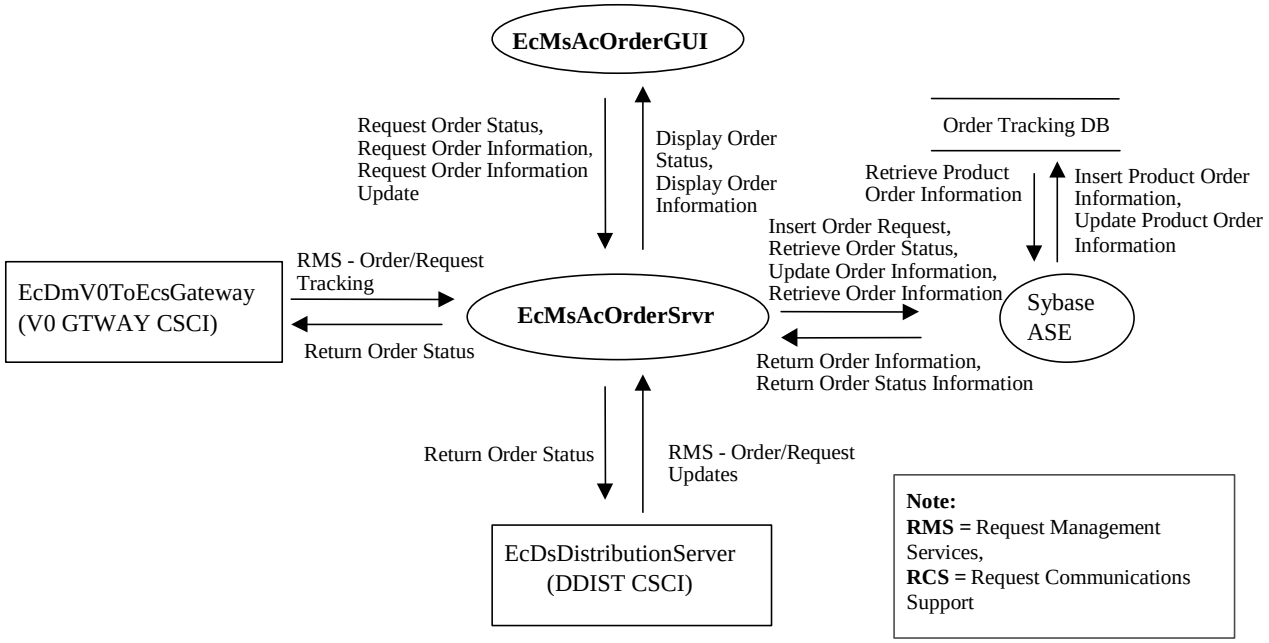


Figure 4.9-7. Accountability Management Service Architecture Diagram

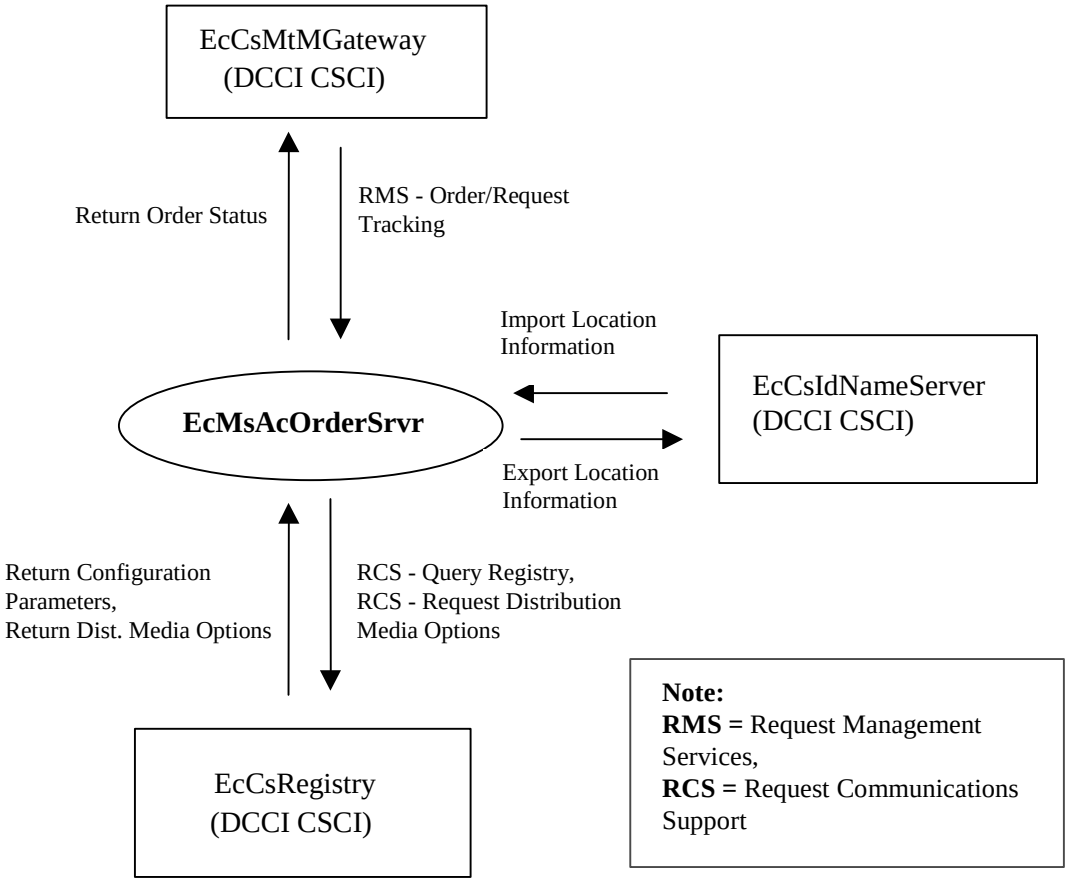


Figure 4.9-7. Accountability Management Service Architecture Diagram (cont.)

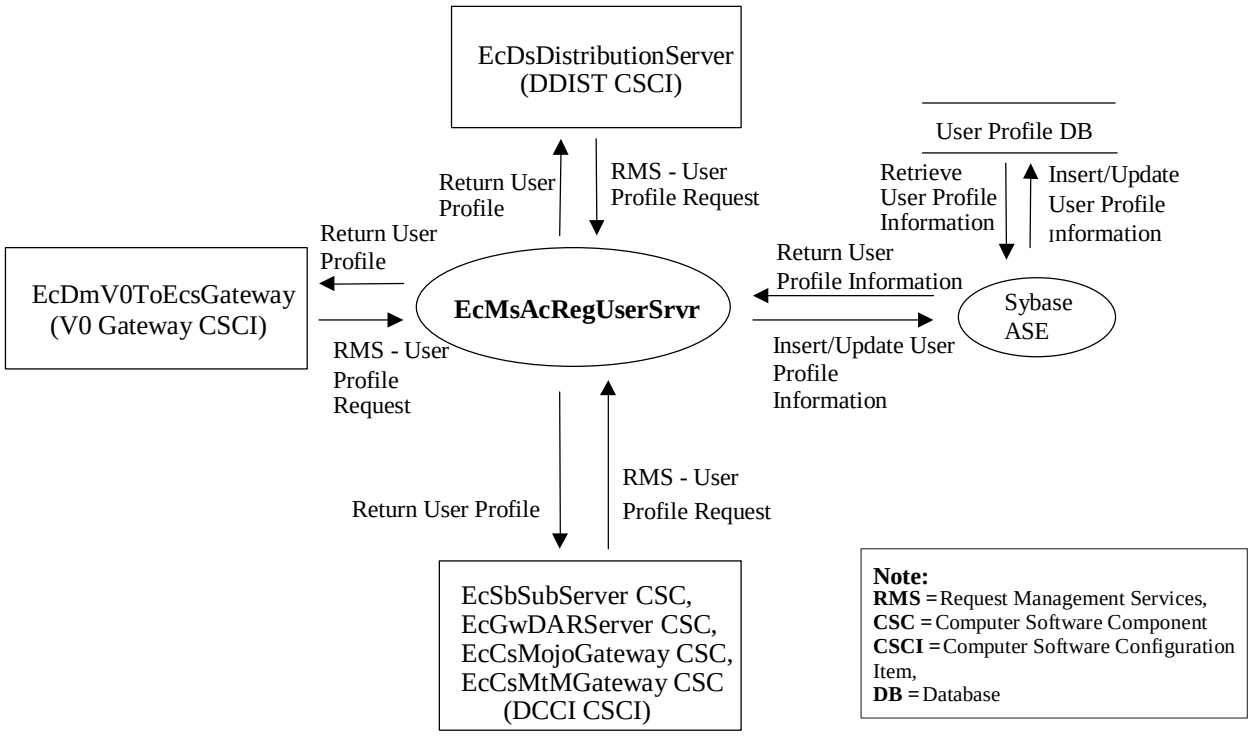


Figure 4.9-7. Accountability Management Service Architecture Diagram (cont.)

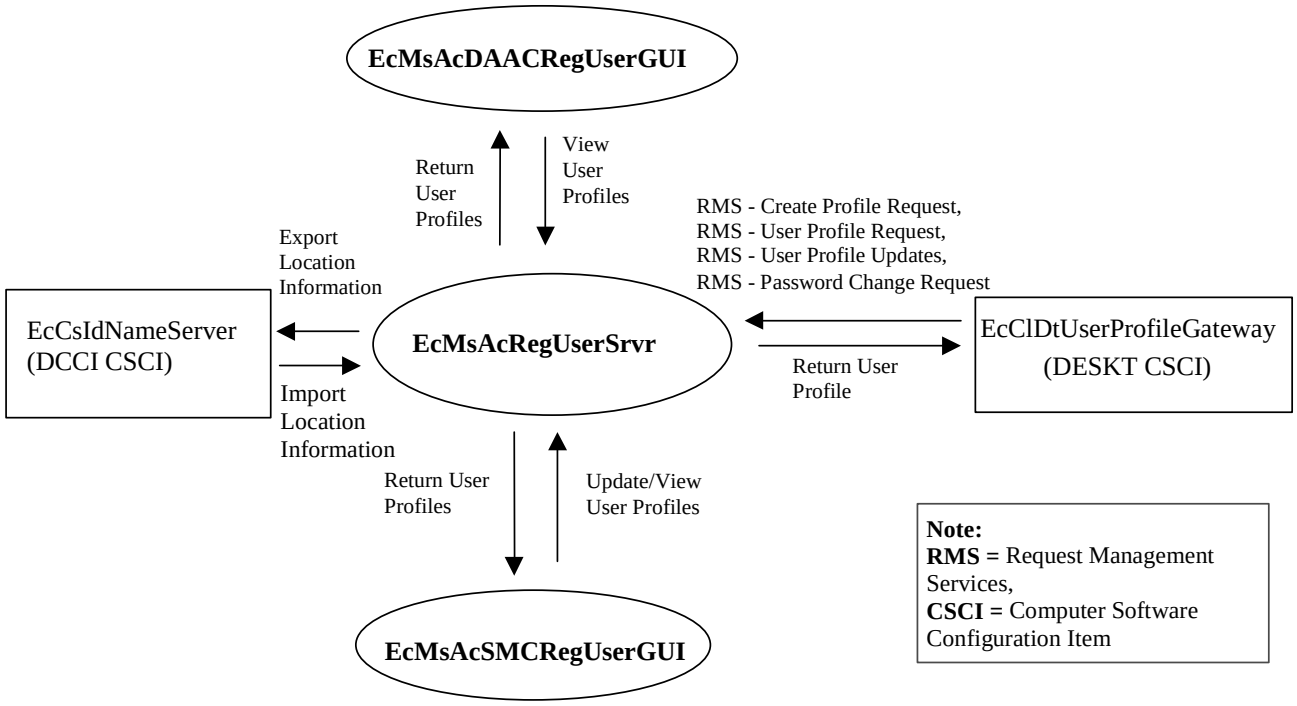


Figure 4.9-7. Accountability Management Service Architecture Diagram (cont.)

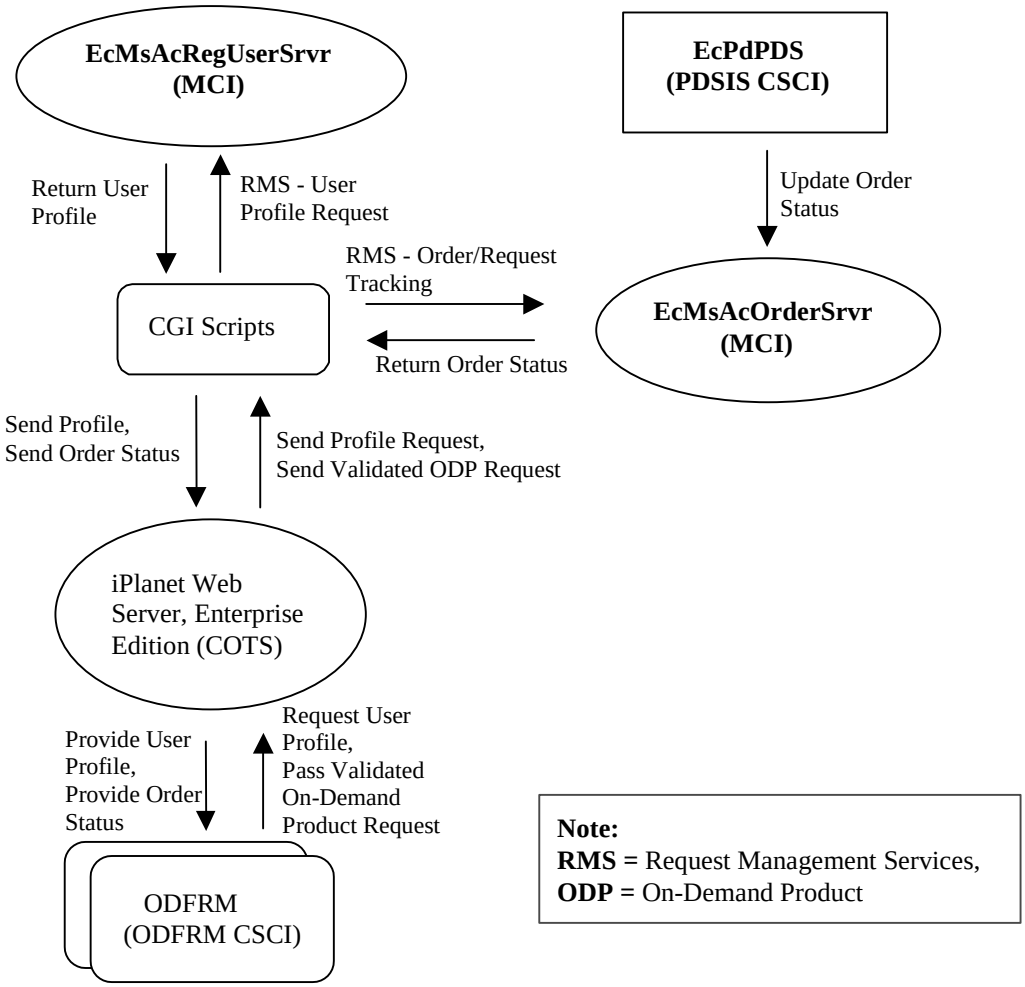


Figure 4.9-7. Accountability Management Service Architecture Diagram (cont.)

4.9.1.1.3.4 Accountability Management Service Processes

Table 4.9-11 provides the descriptions of the processes shown in the Accountability Management Service architecture diagram.

Table 4.9-11. Accountability Management Service Processes (1 of 2)

Process	Type	COTS / Developed	Functionality
EcMsAcRegUserSrvr	Server	Developed	The User Registration Server provides an internal interface to the User Registration GUI and an external interface to other CSCIs/CSCs. The functions are: 1. Insert, delete, update, retrieve registered user profile 2. Retrieve a list of registered user profiles 3. Change V0 gateway password The EcMsAcRegUserSrvr supports: • Single requests at a time • Multiple concurrent requests • Asynchronous request processing • Multiple threads within a single request
EcMsAcSMCRegUserGUI	GUI	Developed	The User Registration graphical user interface enables the viewing and updating of user profiles. The GUI enables the user to: 1. Delete an ECS user 2. Modify an ECS user profile 3. Change the V0 gateway password 4. Change ASTER category and send e-mail 5. Change the DAR privilege 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.
EcMsAcDAACRegUserGUI	GUI	Developed	The User Registration graphical user interface enables the viewing of user profiles. The GUI enables the user to list and view ECS user profiles.

Table 4.9-11. Accountability Management Service Processes (2 of 2)

Process	Type	COTS / Developed	Functionality
EcMsAcOrderSrvr	Server	Developed	The Order Tracking Server provides an external interface to other CSCIs/CSCs. The functions are: 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: • Single requests at a time • Multiple concurrent requests • Asynchronous request processing • Multiple threads within a single request
EcMsAcOrderGUI	GUI	Developed	This graphical user interface enables the user to retrieve the order and request from the Accountability database. The following functions are available: 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 ASE	Server	COTS	The Sybase ASE supports access to the Sybase ASE DBMS. The interface between processes and the databases for storage and retrieval of data or information.

4.9.1.1.3.5 Accountability Management Service Process Interface Descriptions

Table 4.9-12 provides descriptions of the interface events shown in the Accountability Management Service architecture diagram.

Table 4.9-12. Accountability Management Service Process Interface Events (1 of 17)

Event	Event Frequency	Interface	Initiated By	Event Description
Display Order Status	One per order request	*Process:* EcMsAcOrderGUI *Libraries:* MsAcClnt, MsAcComm	*Process:* EcMsAcOrderSrvr *Libraries:* 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	*Process:* EcMsAcOrderGUI *Libraries:* MsAcClnt, MsAcComm	*Process:* EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	The EcMsAcOrderSrvr provides the order information to the EcMsAcOrderGUI for display by the Operations Staff.
Insert Product Order Information	One per order request	*Data Store:* Order Tracking DB	*Process:* Sybase ASE (COTS)	The **Sybase ASE** inserts product order information in the Order Tracking DB.
Update Product Order Information	One per order request	*Data Store:* Order Tracking DB	*Process:* Sybase ASE (COTS)	The **Sybase ASE** updates product order information in the Order Tracking DB.
Retrieve Product Order Information	One per order request	*Data Store:* Order Tracking DB	*Process:* Sybase ASE (COTS)	The **Sybase ASE** retrieves product order information in the Order Tracking DB.
Insert Order Request	One per insert order request	Sybase ASE (COTS)	*Process:* EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	The EcMsAcOrderSrvr submits a request to the **Sybase ASE** to insert a product order request into the Order tracking database (DB).
Retrieve Order Status	One per product order	Sybase ASE (COTS)	*Process:* EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	The EcMsAcOrderSrvr sends a request to the **Sybase ASE** to retrieve the status of an order placed by a user.

Table 4.9-12. Accountability Management Service Process Interface Events (2 of 17)

Event	Event Frequency	Interface	Initiated By	Event Description
Update Order Information	One per update of order information	Sybase ASE (COTS)	*Process:* EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	The EcMsAcOrderSrvr submits a request to the **Sybase ASE** to update order information in the Order tracking database (DB).
Retrieve Order Information	One per update of order information	Sybase ASE (COTS)	*Process:* EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	The EcMsAcOrderSrvr submits a request to the **Sybase ASE** to retrieve order information from the Order tracking database (DB).
Return Order Information	One per return of order information	*Process:* EcMsAcOrder Srvr *Libraries:* MsAcClnt, MsAcComm	*Process:* Sybase ASE (COTS)	The **Sybase ASE** returns product order information per operations request to the EcMsAcOrderSrvr.
Return Order Status Information	One per return order status	*Process:* EcMsAcOrder Srvr *Libraries:* MsAcClnt, MsAcComm	*Process:* Sybase ASE (COTS)	The **Sybase ASE** returns product order status per operations request to the EcMsAcOrderSrvr.
Request Management Services (RMS)	One per service request	N/A	N/A	The EcMsAcOrderSrvr 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 includes the items below.

Table 4.9-12. Accountability Management Service Process Interface Events (3 of 17)

Event	Event Frequency	Interface	Initiated By	Event Description
(RMS – cont.)	At system startup or shutdown and for restarts	*Processes:* EcMsAcOrderSrvr	DAAC unique startup scripts	**System startup and shutdown -** Please refer to the release-related, current version of the Mission Operations Procedures for the ECS Project document (611) and the current ECS Project Training Material document (625), identified in Section 2.2.1 of this document.
(RMS – cont.)	One order per request	*Process:* EcMsAcOrderSrvr *Library:* MsAcClnt *Class:* EcAcOrderCMgr	***DMS Process:*** EcDmV0ToEcsGateway *Class:* DmGwRequestReceiver ***DSS Process:*** EcDsDistributionServer *Library:* DsDdSSh *Class:* DsDdMedia	**Order/Request Tracking** - The **EcDmV0ToEcsGateway** interfaces with the MCI Order/Request Tracking service to create a user product order and track the order. **Order/Request Updates -** The **EcDsDistributionServer** can also create a user product order and retrieve the order by id from the EcMsAcOrderSrvr for update or status.

Table 4.9-12. Accountability Management Service Process Interface Events (4 of 17)

Event	Event Frequency	Interface	Initiated By	Event Description
Return Order Status	One per user request	***DMS Process:*** EcDmV0ToEcsGateway *Class:* DmGwRequestReceiver ***DSS Process:*** EcDsDistributionServer *Library:* DsDdSSh *Class:* DsDdMedia	*Process:* EcMsAcOrderSrvr *Library:* MsAcClnt *Class:* EcAcOrderCMgr	The **EcDmV0ToEcsGateway and EcDsDistributionServer** processes receive the order id and status from the EcMsAcOrderSrvr.
Request Order Status	One per order request	*Process:* EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	*Process:* EcMsAcOrderGUI *Libraries:* MsAcClnt, MsAcComm	The EcMsAcOrderGUI sends requests for order status to the EcMsAcOrderSrvr.
Request Order Information	One per order request	*Process:* EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	*Process:* EcMsAcOrderGUI *Libraries:* MsAcClnt, MsAcComm	The EcMsAcOrderGUI sends requests for order information to the EcMsAcOrderSrvr.
Request Order Information Update	One per order request	*Process:* EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	*Process:* EcMsAcOrderGUI *Libraries:* MsAcClnt, MsAcComm	The EcMsAcOrderGUI sends requests for order information updates to the EcMsAcOrderSrvr.

Table 4.9-12. Accountability Management Service Process Interface Events (5 of 17)

Event	Event Frequency	Interface	Initiated By	Event Description
Request Management Services (RMS)	One per service request	N/A	N/A	The EcMsAcOrderSrvr 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 includes the items below.
(RMS – cont.)	One order per SIPS user request	*Process:* EcMsAcOrderSrvr *Library:* MsAcClnt *Class:* EcAcOrderCMgr	*Processes:* EcCsMtMGateway, EcMsAcOrderSrvr *Library:* MsAcClnt, MsAcComm	**Order/Request Tracking** - The **MTMGW CSC** interfaces with the MCI Order/Request Tracking service to create a user product order and track the progress of the order.
Import Location Information	As required for processing	*Process:* EcCsIdNameServer *Libraries:* EcPf, Middleware, FoNs, FoIp, oodce *Classes:* EcPfManagedServer, CCSMdwNameServer, FoNsNameServerProxy, CCSMdwRwNetProxy	*Process:* EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	The EcMsAcOrderSrvr requests physical and logical server location information from the **EcCsIdNameServer**.

Table 4.9-12. Accountability Management Service Process Interface Events (6 of 17)

Event	Event Frequency	Interface	Initiated By	Event Description
Export Location Information	Once at system startup and after each restart	*Process:* EcCsIdNameServer *Libraries:* EcPf, Middleware, FoNs, FoIp, oodce *Classes:* EcPfManagedServer, CCSMdwNameServer, FoNsNameServerProxy, CCSMdwRwNetProxy	*Process:* EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	The EcMsAcOrderSrvr stores physical and logical server location information in the **EcCsIdnameServer**.

Table 4.9-12. Accountability Management Service Process Interface Events (7 of 17)

Event	Event Frequency	Interface	Initiated By	Event Description
Request Communications Support	Request service(s) as required	***Process:*** EcCsIdNameServer ***Libraries:*** EcPf, Middleware, FoNs, FoIp, oodce *Classes:* EcPfManagedServer, CCSMdwNameServer, FoNsNameServerProxy, CCSMdwRwNetProxy ***Library (Common):*** EcUr *Class:* EcUrServerUR ***Library:*** event *Class:* EcLgErrorMsg ***Process:*** EcCsRegistry *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	*Process:* EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	The **DCCI CSCI** provides a library of services available to each SDPS and CSMS CSCI/CSC. The CSCI services required to perform specific assignments are requested from the DCCI CSCI. These services include: • CCS Middleware Support • Database Connection Services • Name/Address Services • Password Services • Server Request Framework (SRF) • Universal Reference (UR) • Error/Event Logging • Fault Handling Services • Mode Information • Query Registry - Retrieving the requested configuration attribute-value pairs from the Configuration Registry • Request Distribution Media Options from the Configuration Registry

Table 4.9-12. Accountability Management Service Process Interface Events (8 of 17)

Event	Event Frequency	Interface	Initiated By	Event Description
Return Configuration Parameters	One set per request	*Process:* EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	*Process:* EcCsRegistry *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	The **EcCsRegistry** returns the attribute-value pairs (configuration parameters) to the EcMsAcOrderSrvr upon request.
Return Dist. Media Options	One set per request	*Process:* EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	*Process:* EcCsRegistry *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	The **EcCsRegistry** returns the requested distribution media options to the EcDmV0ToEcsGateway.
Return Order Status	One per user request	*Process:* EcCsMtMGateway *Class:* EcCsMtMAcctSrvMgr	*Processes:* EcCsMtMGateway, EcMsAcOrderSrvr *Library:* MsAcClnt, MsAcComm	The **EcCsMtMGateway** receives the order id and status from the EcMsAcOrderSrvr.
Request Management Services (RMS)	One per service request	N/A	N/A	The 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 includes the items below.
(RMS – cont.)	At system startup or shutdown and for restarts	*Processes:* EcMsAcRegUserSrvr,	DAAC unique startup scripts	**System startup and shutdown -** Please refer to the release-related, current version of the Mission Operations Procedures for the ECS Project document (611) and the current ECS Project Training Material document (625), identified in Section 2.2.1 of this document.

Table 4.9-12. Accountability Management Service Process Interface Events (9 of 17)

Event	Event Frequency	Interface	Initiated By	Event Description
(RMS – cont.)	One profile per request per session	*Process:* EcMsAcRegUserSrvr, *Libraries:* MsAcClnt, MsAcComm *Class:* EcAcProfileMgr	***CSS Process:*** EcGwDARServer *Class:* EcGwDARGatewayRequest_S ***CSS Process:*** EcSbSubServer *Class:* EcSbSr ***CSS Process:*** EcCsMojoGateway *Library:* EcCsMojoGateway *Class:* EcMjRetrieveProfileProxy ***DMS Process:*** EcDmV0ToEcsGateway *Class:* DmGwRequestReceiver ***DSS Process:*** EcDsDistributionServer *Library:* DsDdSSh *Class:* DsDdMedia	**User Profile Request** - The EcMsAcRegUserSrvr provides requesting processes (**EcGwDARServer, EcSbSubServer, EcCsMojoGateway, EcDmV0ToEcsGateway,** and **EcDsDistributionServer**) with user profile information such as e-mail address and shipping address to support their processing activities.
Insert/Update User Profile Information	One insert/update per request	*Data Store:* User Profile DB	*Process:* Sybase ASE (COTS)	The **Sybase ASE** stores or updates user profile information by request.
Retrieve User Profile Information	One per profile request	Sybase ASE (COTS)	*Data Store:* User Profile DB	The Sybase ASE obtains the user profile information requested by the M&O Staff from the User Profile DB.

Table 4.9-12. Accountability Management Service Process Interface Events (10 of 17)

Event	Event Frequency	Interface	Initiated By	Event Description
Return User Profile Information	One per profile request	*Process:* EcMsAcRegUserSrvr *Libraries:* MsAcClnt, MsAcComm	Sybase ASE (COTS)	The **Sybase ASE** returns the user profile information to the EcMsAcRegUserSrvr to send back to the user.
Insert/Update User Profile Information	One per user insert/ update profile	Sybase ASE (COTS)	*Process:* EcMsAcRegUserSrvr *Libraries:* MsAcClnt, MsAcComm	The EcMsAcRegUserSrvr sends requests to the **Sybase ASE** to add or modify user profile data in the User Profile database (DB).

Table 4.9-12. Accountability Management Service Process Interface Events (11 of 17)

Event	Event Frequency	Interface	Initiated By	Event Description
Return User Profile	One profile per request	**DSS Process:** EcDsDistributionServer *Library:* DsDdSSh *Class:* DsDdMedia **DMS Process:** EcDmV0ToEcsGateway *Class:* DmGwRequestReceiver **CSS Process:** EcSbSubServer *Class:* EcSbSr, **CSS Process:** EcGwDARServer *Class:* EcGwDARGatewayRequest_S, **CSS Process:** EcCsMojoGateway *Library:* EcCsMojoGateway *Class:* EcMjRetrieveProfileProxy, **CSS Process:** EcCsMtMGateway *Class:* EcCsMtMAcctSrvMgr	*Process:* EcMsAcRegUserSrvr, *Libraries:* MsAcClnt, MsAcComm *Class:* EcAcProfileMgr	The EcMsAcRegUserSrvr returns user profile information provided from the Sybase ASE to the requester (**EcDsDistributionServer**, **EcDmV0ToEcsGateway**, **EcSbSubServer**, **EcGwDARServer**, **EcCsMojoGateway** and **EcCsMtMGateway**).

Table 4.9-12. Accountability Management Service Process Interface Events (12 of 17)

Event	Event Frequency	Interface	Initiated By	Event Description
View User Profiles	One per view user profile request	*Process:* EcMsAcRegUserSrvr *Libraries:* MsAcClnt, MsAcComm	M&O staff *Process:* EcMsAcDAACRegUserGUI *Libraries:* MsAcClnt, MsAcComm	The **M&O staff** request, via the EcMsAcDAACRegUserGUI, to retrieve a user profile from the EcMsAcRegUserSrvr.
Request Management Services (RMS)	One per service request	N/A	N/A	The EcMsAcRegUserSrvr 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 includes the items below.
(RMS - cont.)	Per user request	*Process:* EcMsAcRegUserSrvr, *Libraries:* MsAcClnt, MsAcComm *Class:* EcAcProfileMgr	*CLS Process:* EcClDtUserProfileGateway *Class:* ClDtProfileServer	**Create Profile Request -** The **EcClDtUserProfileGateway** sends user information for becoming a registered user of the ECS to the EcMsAcRegUserSrvr. The EcMsAcRegUserSrvr sends a response to the user when the request is received.
(RMS – cont.)	One profile per request per session	*Process:* EcMsAcRegUserSrvr, *Libraries:* MsAcClnt, MsAcComm *Class:* EcAcProfileMgr	***CLS Process:*** EcClDtUserProfileGateway *Class:* ClDtProfileServer	**User Profile Request** - The EcMsAcRegUserSrvr provides the requesting process (**EcClDtUserProfileGateway**) with user profile information such as e-mail address and shipping address to support their processing activities.

Table 4.9-12. Accountability Management Service Process Interface Events (13 of 17)

Event	Event Frequency	Interface	Initiated By	Event Description
(RMS – cont.)	One profile per request per session One password change per configured period	*Process:* EcMsAcRegUserSrvr, *Libraries:* MsAcClnt, MsAcComm *Class:* EcAcProfileMgr	***CLS Process:*** EcClDtUserProfileGateway *Class:* ClDtProfileServer	**User Profile Updates** - The EcMsAcRegUserSrvr provides the requesting process (**EcClDtUserProfileGateway**) with updates to user profile parameters such as e-mail address and shipping address to support their processing activities. **Password Change Request -** The **EcClDtUserProfileGateway** sends requests on behalf of ECS users to the EcMsAcRegUserSrvr to change or reset users' authenticators in the MSS database.
Return User Profile	One per request	Process: EcClDtUserProfileGateway *Class:* ClDtProfileServer	*Process:* EcMsAcRegUserSrvr, *Libraries:* MsAcClnt, MsAcComm *Class:* EcAcProfileMgr	The EcMsAcRegUserSrvr returns user profile information provided from the Sybase ASE to the requester (**EcClDtUserProfileGateway**).
Update/View User Profiles	One per update/view user profile	*Process:* EcMsAcRegUserSrvr *Libraries:* MsAcClnt, MsAcComm	M&O staff *Process:* EcMsAcSMCRegUserGUI *Libraries:* MsAcClnt, MsAcComm *Class:* MsAcRegUsrUtl	The **M&O staff** sends requests to modify or view user profile information via the EcMsAcSMCRegUserGUI.
Return User Profiles	One per return of user profile	*Processes:* EcMsAcSMCRegUserGUI, EcMsAcDAACRegUserGUI *Libraries:* MsAcClnt, MsAcComm	*Process:* EcMsAcRegUserSrvr *Libraries:* MsAcClnt, MsAcComm	The EcMsAcRegUserSrvr returns user profile information provided from the **Sybase ASE** to the requester at the EcMsAcSMCRegUserGUI or EcMsAcDAACRegUserGUI.

Table 4.9-12. Accountability Management Service Process Interface Events (14 of 17)

Event	Event Frequency	Interface	Initiated By	Event Description
Import Location Information	As required for processing	*Process:* EcCsIdNameServer *Libraries:* EcPf, Middleware, FoNs, FoIp, oodce *Classes:* EcPfManagedServer, CCSMdwNameServer, FoNsNameServerProxy, CCSMdwRwNetProxy	*Process:* EcMsAcRegUserSrvr *Libraries:* MsAcClnt, MsAcComm	The EcMsAcRegUserSrvr requests physical and logical server location information from the **EcCsIdNameServer**.
Export Location Information	Once at system startup and after each restart	*Process:* EcCsIdNameServer *Libraries:* EcPf, Middleware, FoNs, FoIp, oodce *Classes:* EcPfManagedServer, CCSMdwNameServer, FoNsNameServerProxy, CCSMdwRwNetProxy	*Process:* EcMsAcRegUserSrvr *Libraries:* MsAcClnt, MsAcComm	The EcMsAcRegUserSrvr stores physical and logical server location information in the **EcCsIdnameServer**.

Table 4.9-12. Accountability Management Service Process Interface Events (15 of 17)

Event	Event Frequency	Interface	Initiated By	Event Description
Request Management Services (RMS	One per user request	*Process:* EcMsAcRegUserSrvr, *Libraries:* MsAcClnt, MsAcComm *Class:* EcAcProfileMgr	*Processes:* CGI scripts	**User Profile Request** - The EcMsAcRegUserSrvr receives requests from **CGI scripts** for user profile information such as e-mail address and shipping address to support their processing activities. **Order/Request Tracking** - The ODFRM CSCI, via the iPlanet Web Server, uses **CGI scripts** to interface with the MCI Order/Request Tracking service to create a user product order.
Update Order Status	One per product request	*Process:* EcMsAcOrderSrvr (By executing stored procedure: ProcUpdPdsRequest)	*Process:* EcPdPDSISServer *Libraries:* Pdsis.jar, ECS package *Class:* EcsMssUpdate	The **EcPdPDS** sends a request to update the order status to the EcMsAcOrderSrvr.
Return Order Status	One per user request	*Processes:* CGI scripts	*Process:* EcMsAcOrderSrvr *Library:* MsAcClnt *Class:* EcAcOrderCMgr	The EcMsAcOrderSrvr returns an order id and status to the CGI scripts on behalf of a user for a requested ECS product.
Send Profile Request	One request per user	*Processes:* CGI scripts	*Process:* iPlanet Web Server, Enterprise Edition (COTS)	The iPlanet Web Server sends a request for a user profile to the CGI scripts on behalf of a user.

Table 4.9-12. Accountability Management Service Process Interface Events (16 of 17)

Event	Event Frequency	Interface	Initiated By	Event Description
Send Validated ODP Request	One request per user	*Processes:* CGI scripts	*Process:* iPlanet Web Server, Enterprise Edition(COTS)	The iPlanet Web Server validates and sends a on-demand product request to the CGI scripts.
Request User Profile	One per user	*Process:* iPlanet Web Server, Enterprise Edition (COTS)	*Process:* EcClOdRequest *Library:* ClOdCommon *Class:* ClOdProductRequest	The ODFRM sends a request to the iPlanet Web Server for a user profile for the user to access ECS products.
Pass Validated On-Demand Product Request	One request per user	*Process:* iPlanet Web Server, Enterprise Edition (COTS)	*Process:* EcClOdRequest *Library:* ClOdCommon *Class:* ClOdProductRequest	The ODFRM sends an on-demand product request to the iPlanet Web Server for validation and forwarding to the EcMsAcOrder Srvr.
Provide User Profile	One per user	*Process:* EcClOdRequest *Library:* ClOdCommon *Class:* ClOdProductRequest	*Process:* iPlanet Web Server, Enterprise Edition (COTS)	The iPlanet Web Server returns the user profile to the ODFRM.
Provide Order Status	One per request	*Process:* EcClOdRequest *Library:* ClOdCommon *Class:* ClOdProductRequest	*Process:* iPlanet Web Server, Enterprise Edition (COTS)	The iPlanet Web Server returns the order id and status for the requested ECS product(s) to the ODFRM.
Send Profile	One per user	*Process:* iPlanet Web Server, Enterprise Edition (COTS)	*Processes:* CGI scripts	The CGI scripts provide the user profile to the iPlanet Web Server.

Table 4.9-12. Accountability Management Service Process Interface Events (17 of 17)

Event	Event Frequency	Interface	Initiated By	Event Description
Send Order Status	One per request	*Process:* iPlanet Web Server, Enterprise Edition (COTS)	*Processes:* CGI scripts	The CGI scripts provide an order id and status for the requested ECS product to the iPlanet Web Server.
Return User Profile	One per user	*Processes:* CGI scripts	*Process:* EcMsAcRegUserSrvr, *Libraries:* MsAcClnt, MsAcComm *Class:* EcAcProfileMgr	The EcMsAcRegUserSrvr returns a user profile to the CGI scripts for a user to access or use ECS products or services.

4.9.1.1.3.6 Accountability Management Service Data Stores

Table 4.9-13 provides descriptions of the data stores shown in the Accountability Management Service architecture diagram.

Table 4.9-13. Accountability Management Service Data Stores

Data Store	Type	Description
User Profile DB	Database	The User Profile DB 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.

4.9.1.1.4 MCI - Trouble Ticket Computer Software Component Description

4.9.1.1.4.1 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.

4.9.1.1.4.2 Trouble Ticket Context

Figure 4.9-8 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, NISN, 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. System-level TTs are forwarded to the SMC for resolution by the EMD staff. TTs that cannot be resolved in a specified amount of time are converted to NCR format and submitted (via email) to the EDF's Distributed Defect Tracking System (ClearDDTS) for NCR processing.

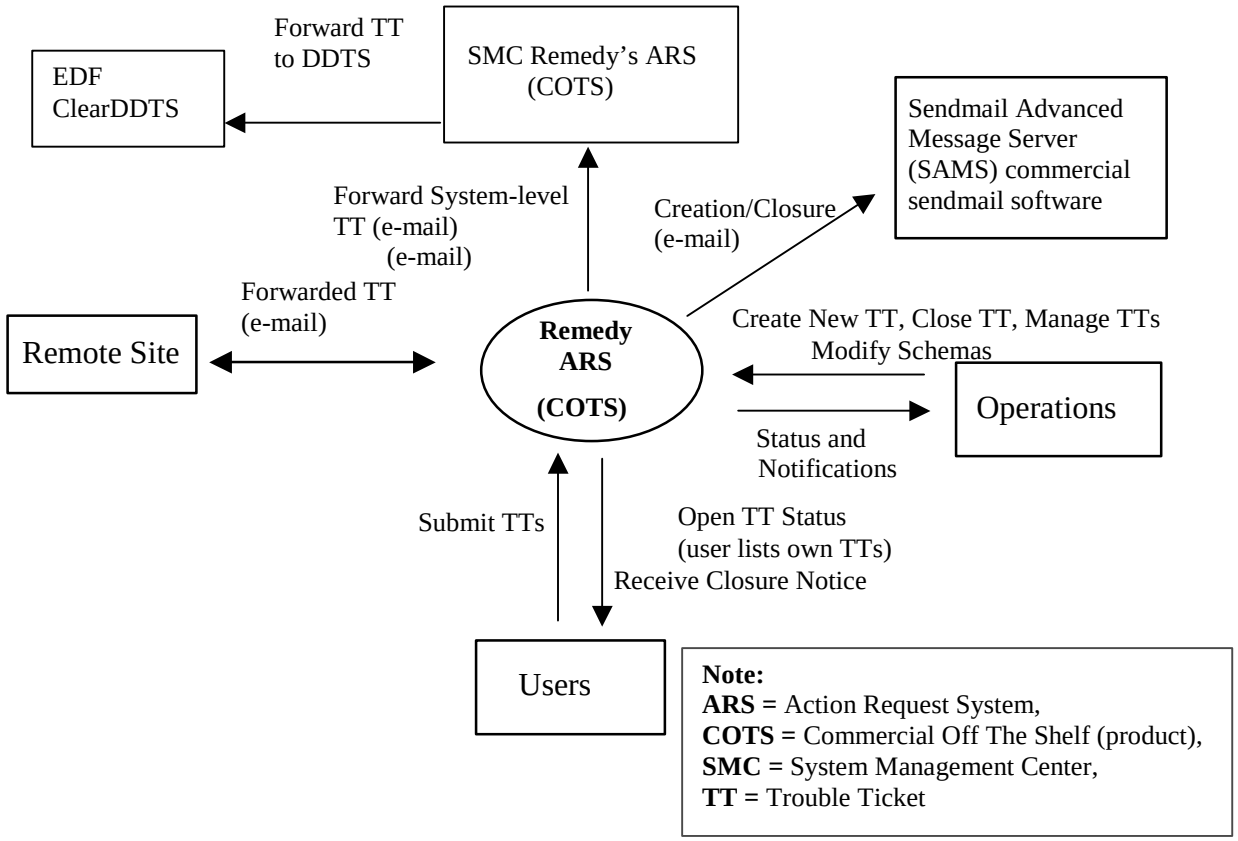


Figure 4.9-8. Trouble Ticket Context Diagram

Table 4.9-14 provides descriptions of the interface events shown in the Trouble Ticket context diagram.

Table 4.9-14. Trouble Ticket Interface Events

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 via the **Sendmail Advanced Message Server (SAMS) commercial sendmail software**.
Create New TT	The M&O staff (**Operations**) is 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 (**Operations**) 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 (**Operations**) 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.
Modify Schemas	The TT administrator (**Operations**) 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 (**Operations**) 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 (**Users**) 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 (**Users**) after the TT has been closed. This message includes the TT ID number and corrective actions taken.
Submit TTs	An ECS user (**Users**) 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 (**Remote Site**). This new TT has the original ID stored as a Unique Identifier.
Forward TT to SMC (e-mail)	Using the Forward TT capability, the site administrator can select and forward a copy of a TT to the **SMC** (**Remedy's ARS**). The original TT's ID is stored as the ticket's unique identifier.
Forward TT to DDTS	Using the Remedy TT conversion to NCR format capability at the SMC, the EMD staff can transmit the contents of a TT to the EDF's DDTS COTS application.

4.9.1.1.4.3 Trouble Ticket Architecture

Figure 4.9-9 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 ASE).

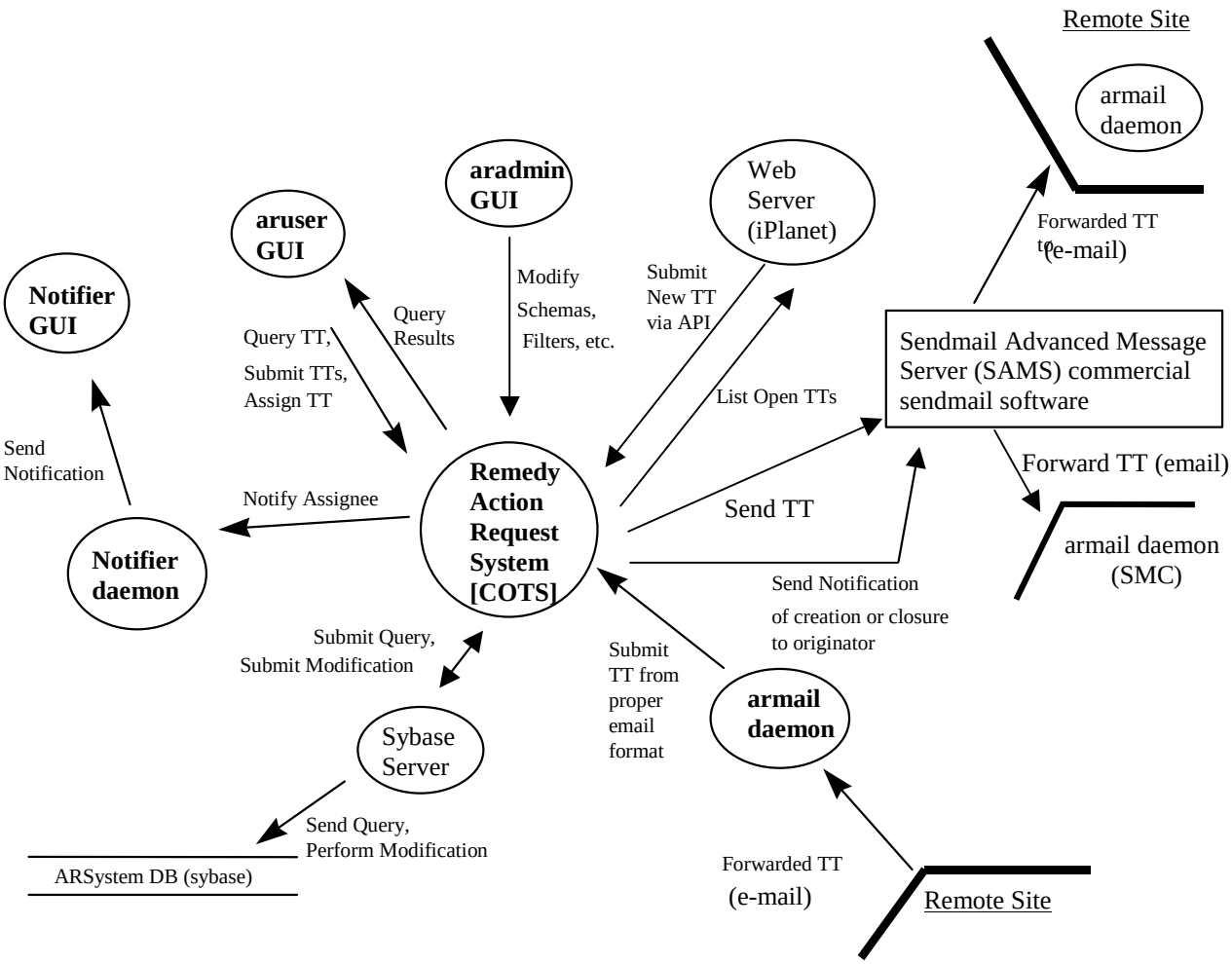


Figure 4.9-9. Trouble Ticket Architecture Diagram

4.9.1.1.4.4 Trouble Ticket Process Descriptions

Table 4.9-15 provides descriptions of the processes shown in the Trouble Ticket architecture diagram.

Table 4.9-15. Trouble Ticket Processes

Process	Type	COTS / Developed	Functionality
aruser GUI	GUI	COTS	The aruser GUI enables the M&O staff member to: 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 administrator uses this GUI to: 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: 1. Update schemas and aruser screen layouts 2. Update escalation policies and filters
Web Server (iPlanet)	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: 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.
Sendmail Advanced Message Server (SAMS) commercial sendmail software	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 ASE	Server	COTS	Sybase ASE accepts queries and requests for modifications to data in persistent storage in the ARSystem DB from the Remedy ARS.

4.9.1.1.4.5 Trouble Ticket Process Interface Descriptions

Table 4.9-16 provides descriptions of the interface events shown in the Trouble Ticket architecture diagram.

Table 4.9-16. 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.	*COTS:* Remedy ARS *Program:* MsTtManager	*Process:* 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	*COTS:* Remedy ARS *Programs:* MsTtHTMLItems, MsTtServiceRequestor, MsTtManager *Classes:* MsAcUsrProfile, MsAcUsrProfileMgr_1_0	User iPlanet Web Server (COTS)	Alternatively, an ECS user can submit a TT via the custom **iPlanet Web Server** 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	IPlanet Web Server (COTS)	User *COTS:* Remedy ARS *Programs:* MsTtHTMLMenu, MsTtHTMLItems, MsTtManager *Classes:* MsAcUsrProfile, MsAcUsrProfileMgr_1_0	The ECS user can also query the Remedy system through the custom **iPlanet Web Server** 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	*Process:* armail daemon (COTS)	*Process:* Sendmail Advanced Message Server (SAMS) commercial sendmail software (COTS)	The M&O staff member sends the TT to another site (**Remote Site**) to be archived or for escalation to a review board for action.

Table 4.9-16. Trouble Ticket Process Interface Events (2 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
Send TT	One per TT sent	*Process:* Sendmail Advanced Message Server (SAMS) commercial sendmail software (COTS)	*COTS:* Remedy ARS *Program:* MsTtManager	Remedy ARS uses e-mail (**SAMS sendmail software**) to send an open or closed TT to the appropriate site.
Send Notification of creation or closure to originator	One per notification sent to originator	Sendmail Advanced Message Server (SAMS) commercial sendmail software (COTS)	*COTS:* Remedy ARS *Program:* MsTtManager	The originators receive notification of creation or closure on their TT via e-mail sent using the **SAMS sendmail software** on the host where Remedy is running.
Submit TT from proper e-mail format	One per submit of TT in proper e-mail format	*COTS:* Remedy ARS *Program:* MsTtManager	*Process:* 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 ASE (COTS)	*COTS:* Remedy ARS *Program:* MsTtManager	After resolving the TT issue, the M&O Staff uses Remedy ARS to query the **Sybase ASE** for the appropriate TT and receives the information from Sybase.
Submit Modification	One per modification	Sybase ASE (COTS)	*COTS:* Remedy ARS *Program:* MsTtManager	The M&O Staff modifies the status (changes the status to Closed) of the appropriate TT via the **Sybase ASE**.

Table 4.9-16. Trouble Ticket Process Interface Events (3 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
Send Query	One per query	*Data Store:* ARSystem DB	Sybase ASE (COTS)	The **Sybase ASE** is used to query data in persistent storage.
Perform Modification	One per update	*Data Store:* ARSystem DB	Sybase ASE (COTS)	The **Sybase ASE** is used to modify data in persistent storage.
Notify Assignee	One per notify assignee	*Process:* Notifier daemon (COTS)	*COTS:* Remedy ARS *Program:* 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	*Process:* Notifier GUI (COTS)	*Process:* Notifier daemon (COTS)	The notifier daemon notifies the responsible M&O staff member via the **notifier GUI** or by e-mail depending on User schema settings.
Query TT	One per TT	*COTS:* Remedy ARS *Program:* MsTtManager	*Process:* aruser GUI (COTS)	The M&O staff member, upon receiving notification of a new TT, via the **aruser GUI**, queries the Remedy TT schema to find the detailed information and process the TT.
Submit TTs	One per submit TT	*Program:* MsTtManager (COTS, Remedy ARS)	M&O staff *Process:* aruser GUI (COTS)	A new Trouble Ticket is created by M&O staff and entered into the Remedy system via the **aruser GUI**. A notification is sent to the TT administrator of the existence of a new TT.
Assign TT	One per assign TT	*Program:* MsTtManager (COTS, Remedy ARS)	M&O staff *COTS:* aruser GUI (COTS)	After receiving the notification from the **aruser GUI**, the TT administrator assigns the TT to an M&O staff member.
Query Results	One per query	*Process:* aruser GUI (COTS)	*COTS:* Remedy ARS *Program:* MsTtManager	TT query results are returned from the Remedy ARS to the M&O staff via the **aruser GUI**.

4.9.1.1.4.6 Trouble Ticket Data Stores

Table 4.9-17 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 4.9-17. 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: 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

4.9.1.1.5 MCI - Networker Backup/Restore Computer Software Component Description

4.9.1.1.5.1 Networker 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.

4.9.1.1.5.2 Networker Backup/Restore Context

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

4.9.1.1.5.3 Networker Backup/Restore Architecture

Figure 4.9-10 is the Networker Backup/Restore architecture diagram. The diagram shows the events sent to the Network Server of the Networker Backup/Restore CSC and the events the Network Server of the Networker Backup/Restore CSC sends to other processes (network clients).

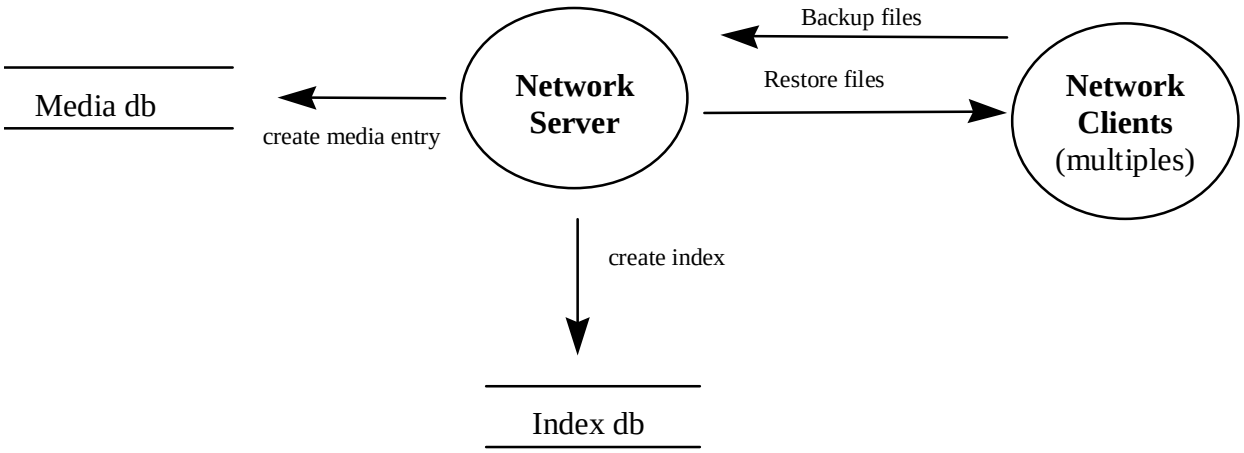


Figure 4.9-10. Networker Backup/Restore Architecture Diagram

4.9.1.1.5.4 Networker Backup/Restore Process Descriptions

Table 4.9-18 provides descriptions of the processes shown in the Network Backup/Restore architecture diagram.

Table 4.9-18. Networker 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.

4.9.1.1.5.5 Networker Backup/Restore Process Interface Descriptions

Table 4.9-19 provides descriptions of the interface events shown in the Networker Backup/Restore architecture diagram.

Table 4.9-19. Networker Backup/Restore Process Interface Events

Event	Event Frequency	Interface	Initiated By	Event Description
Backup Files	One per backup	*Process:* Network Server (COTS)	*Process:* Network Clients	Data is passed from the client to the server and archived to tape.
Restore Files	One per restore	*Process:* Network Clients	*Process:* 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)	*Process:* 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)	*Process:* 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.

4.9.1.1.5.6 Networker Backup/Restore Data Stores

Table 4.9-20 provides descriptions of the data stores shown in the Networker Backup/Restore architecture diagram.

Table 4.9-20. Networker 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.

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

4.9.1.1.6.1 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 (505-41-34, 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 505-41-34, pages 8-4 and 8-5. 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 Monitoring 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).

4.9.1.1.6.2 ASTER E-mail Header Handler Context

Figure 4.9-11 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 4.9-21 provides descriptions of the interface events in the ASTER E-mail Header Handler context diagram.

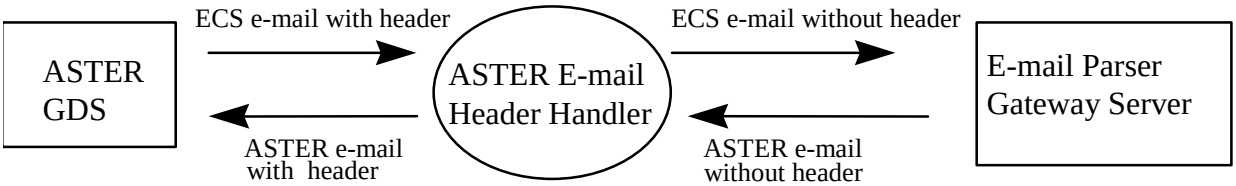


Figure 4.9-11. ASTER E-mail Header Handler Context Diagram

Table 4.9-21. 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 by the **E-mail Parser Gateway Server**.
ASTER e-mail without header	The **E-Mail Parser Gateway Server** sends an e-mail message to a predefined ASTER e-mail alias within the ECS, without a header (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 GDS** 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.

4.9.1.1.6.3 ASTER E-mail Header Handler Architecture

Figure 4.9-12 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 Monitoring Center and the ASTER GDS.

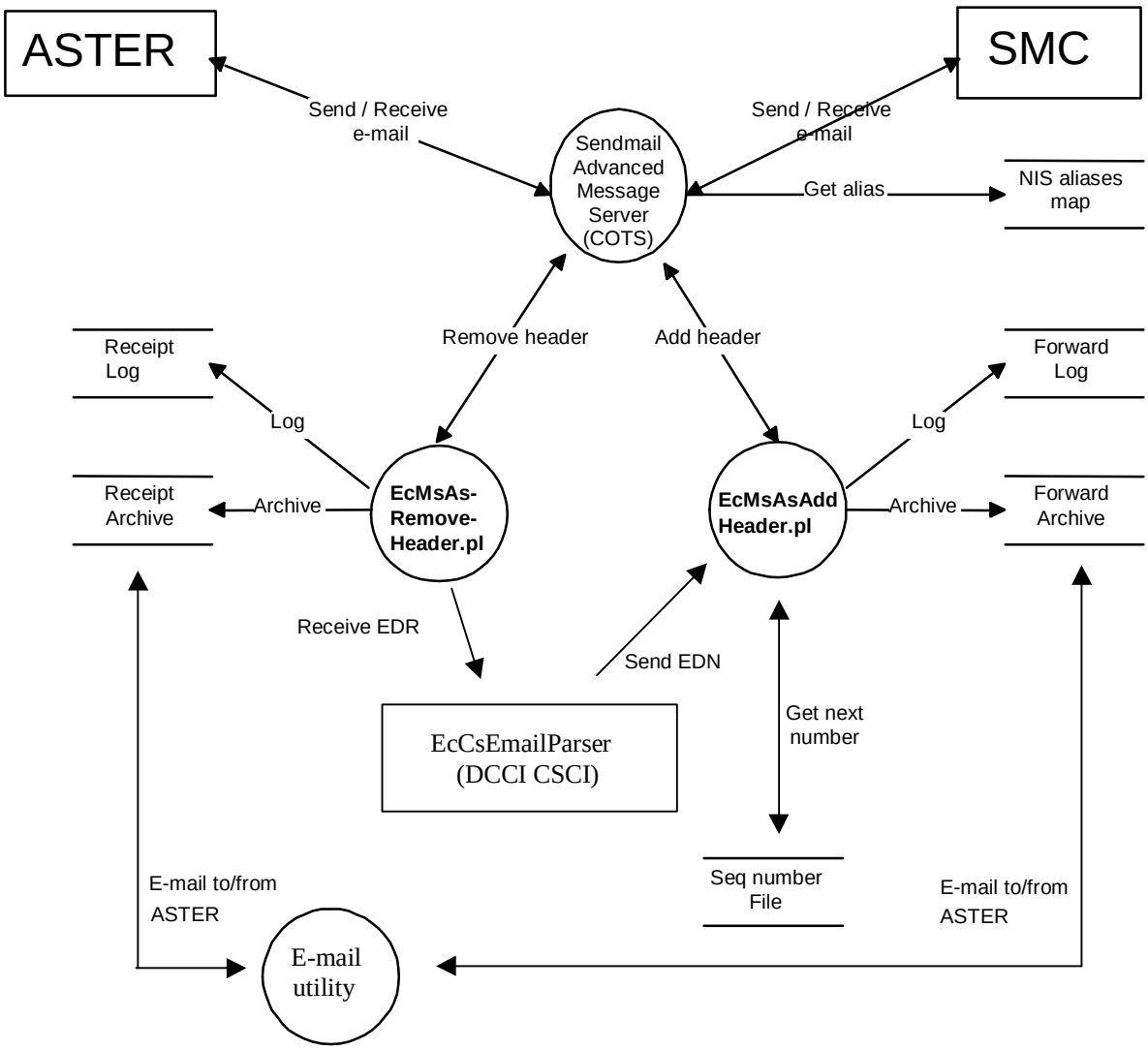


Figure 4.9-12. ASTER E-mail Header Handler Architecture Diagram

4.9.1.1.6.4 ASTER E-mail Header Handler Process Descriptions

Table 4.9-22 provides descriptions of the ASTER E-mail Header Handler processes shown in the ASTER E-mail Header Handler architecture diagram.

Table 4.9-22. ASTER E-mail Header Handler Processes

Process	Type	COTS / Developed	Functionality
Sendmail Advanced Message Server (SAMS) commercial sendmail software	Server	COTS	The sendmail daemon, which is a commercial software package, handles delivery and receipt of e-mail messages.
EcMsAsAddHeader.pl	Script	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	Script	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	Program	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.

4.9.1.1.6.5 ASTER E-mail Header Handler Process Interface Descriptions

Table 4.9-23 provides descriptions of the interface events shown in the ASTER E-mail Header Handler architecture diagram.

Table 4.9-23. ASTER E-mail Header Handler Process Interface Events (1 of 3)

Interface	Event Frequency	Interface	Initiated By	Event Description
Send e-mail	One per e-mail send	SMC, ASTER GDS	Sendmail Advanced Message Server (SAMS) daemon (API, system call or command line) – (COTS)	The **Sendmail Advanced Message Server (SAMS)** daemon attempts to forward e-mail messages to the specified recipient at the SMC or the ASTER GDS.

Table 4.9-23. ASTER E-mail Header Handler Process Interface Events (2 of 3)

Interface	Event Frequency	Interface	Initiated By	Event Description
Receive e-mail	One per e-mail receive	SMC, ASTER GDS	Sendmail Advanced Message Server (SAMS) daemon (SMTP protocols) – (COTS)	The **Sendmail Advanced Message Server (SAMS)** 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	*Data Store:* NIS aliases map	Sendmail Advanced Message Server (SAMS) daemon (NIS system call) - [COTS]	While processing e-mail, the **Sendmail Advanced Message Server (SAMS)** 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	Sendmail Advanced Message Server (SAMS) daemon [COTS]	*Script:* 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 **Sendmail Advanced Message Server (SAMS)** daemon to be sent to its destination.
Log	One per log	*Data Store:* Forward Log *Data Store:* Receipt Log	*Script:* EcMsAsAddHeader.pl *Script:* EcMsAsRemoveHeader.pl	An entry is added to note the e-mail message date, time, and recipient are being forwarded before being stored in the **Receipt Log**.
Archive	One per archive	*Data Store:* Forward Archive *Data Store:* Receipt Archive	*Script:* EcMsAsAddHeader.pl *Script:* 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) before being stored in the **Receipt Archive**.

Table 4.9-23. ASTER E-mail Header Handler Process Interface Events (3 of 3)

Interface	Event Frequency	Interface	Initiated By	Event Description
Send EDN	One per E-mail send	*Script:* EcMsAsAddHeader	*Process:* EcCsEmailParser *Class:* EcCsEmailParser	The **EcCsEmailParser** sends the EDN to the EcMsAsAddHeader to have a header added.
Get next number	One per get of next number	*Data Store:* Seq number File	*Script:* EcMsAsAddHeader.pl	The EcMsAcAddHeader.pl script obtains the next sequence number from a text file (**Seq Number File**).
E-mail to/from ASTER	One per e-mail to/from ASTER	*Data Stores:* Receipt Archive, Forward Archive	*Process:* 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, using an **E-mail utility,** and copies are kept in data files within the ECS.
Receive EDR	One per EDR sent	*Process:* EcCsEmailParser *Class:* EcCsEmailParser	*Script:* 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. The EDR is sent to the **EcCsEmailParser**.
Remove header	One per message	*Process:* EcCsEmailParser *Class:* EcCsEmailParser	*Script:* EcMsAsRemoveHeader.pl	The ASTER header is removed from messages sent for local delivery by the EcMsAsRemoveHeader.pl script.

4.9.1.1.6.6 ASTER E-mail Header Handler Data Stores

Table 4.9-24 provides descriptions of the data stores shown in the ASTER E-mail Header Handler architecture diagram.

Table 4.9-24. ASTER E-mail Header Handler Data Stores

Data Store	Type	Functionality
NIS aliases map	Database table	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.

4.9.1.2 Management Software Context

Figure 4.9-13 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 4.9-25 provides descriptions of the interface events shown in the MCI context diagram.

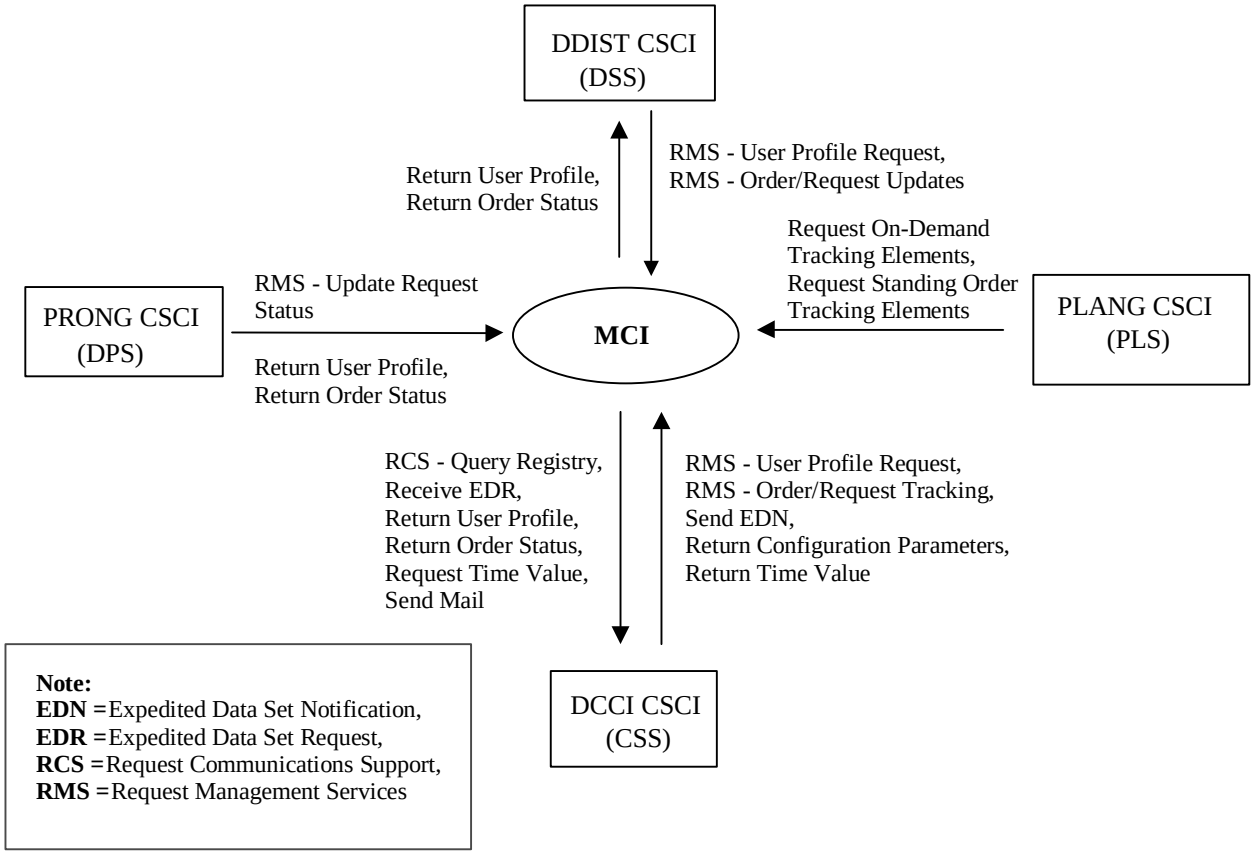


Figure 4.9-13. Management Software CSCI (MCI) Context Diagram

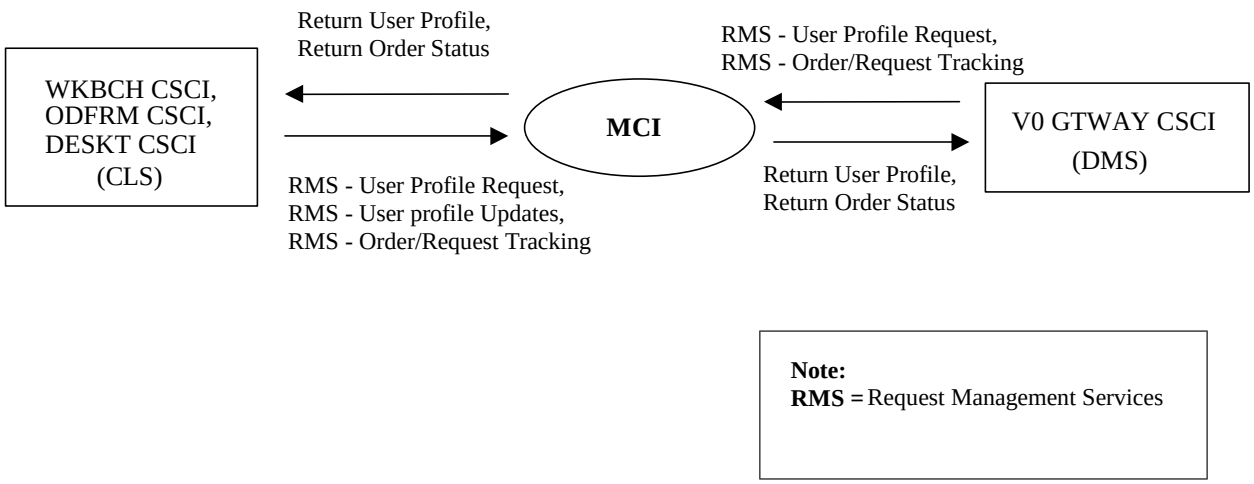


Figure 4.9-13. Management Software CSCI (MCI) Context Diagram (cont.)

Table 4.9-25. Management Software CSCI (MCI) Interface Events (1 of 2)

Event	Interface Event Description
Request Management Services	The MCI 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 includes: The MCI interfaces with other subsystems to perform the following: • **User Profile Request** - The MCI provides requesting CSCIs (**DDIST** and **DCCI**) with User Profile information such as e-mail address and shipping address to support their processing activities. • **Order/Request Updates** - The **DDIST CSCI** interfaces with the Accountability Management Service Order/Request Tracking to create/update the EcAcRequest (user product order request) such as media id, quantity and type. • **Update Request Status** - The **PRONG CSCI** requests the MCI 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 On-Demand Tracking Elements	The **PLANG CSCI** requests on-demand, tracking elements (i.e., Order ID and Request ID) from the MCI.
Request Standing Order Tracking Elements	The **PLANG CSCI** requests standing order tracking elements based upon orders of ASTER high-level products (i.e., Order ID and Request ID) from the MCI.
Send EDN	The **DCCI CSCI** stores the EDN messages with URs, time range, etc., and sends the EDN to the MCI.
Return Configuration Parameters	The **DCCI CSCI** returns the requested configuration parameters to the MCI.
Return Time Value	The **DCCI CSCI** returns a time value to the MCI.

Table 4.9-25. Management Software CSCI (MCI) Interface Events (2 of 2)

Event	Interface Event Description
Request Communications Support (RCS)	The **DCCI CSCI** provides a library of services available to each SDPS and CSMS CSCI/CSC. The CSCI services required to perform specific assignments are requested from the DCCI CSCI. These services include: • CCS Middleware Support • Database Connection Services • File Transfer Services • Network & Distributed File Services • Bulk Data Transfer Services • Name/Address Services • Password Services • Server Request Framework (SRF) • Universal Reference (UR) • Error/Event Logging • Fault Handling Services • Mode Information • Query Registry - Retrieving the requested configuration attribute-value pairs from the Configuration Registry • Request Distribution Media Options from the Configuration Registry
Receive EDR	The MCI strips the EDR message header and the message is sent to the **DCCI CSCI**.
Return User Profile	The MCI returns user profile information to the **DDIST** and **DCCI** CSCIs.
Return Order Status	The MCI returns the status of an order (upon request) to the **DCCI** and **DDIST** CSCIs.
Request Time Value	The MCI submits time requests to the **DCCI CSCI**.
Send Mail	The MCI uses the CsEmMailRelA interface to send mail to the **DCCI CSCI**.
Request Management Services	The MCI 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 includes: The MCI interfaces with other subsystems to perform the following: • **Order/Request Tracking** - The **ODFRM CSCI**, via the iPlanet Web Server, uses CGI scripts to interface with the MCI Order/Request Tracking service to create a user product order and submit the order to the PLS. In addition, the MTMGW CSC (within the **DCCI CSCI**) can also create a user product order and retrieve it from the MCI. The **V0 GTWAY CSCI** interfaces with the MCI Order/Request Tracking service to create a user product order record. • **User Profile Request** - The MCI provides requesting CSCIs (**V0 GTWAY, DESKT** and **WKBCH**) with User Profile information 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 (via the **DESKT CSCI**) and makes the updates in the user profile database.
Return User Profile	The MCI returns user profile information to the **V0 GTWAY** and **DESKT** CSCIs.
Return Order Status	The MCI returns the status of an order (upon request) to the **V0 GTWAY** and **ODFRM** CSCIs.

4.9.1.3 Management Software Architecture

Figure 4.9-14 is the Management Software CSCI (MCI) architecture diagram. The architecture diagram shows the events sent to the MCI processes and the events sent by the MCI processes to other processes or COTS software.

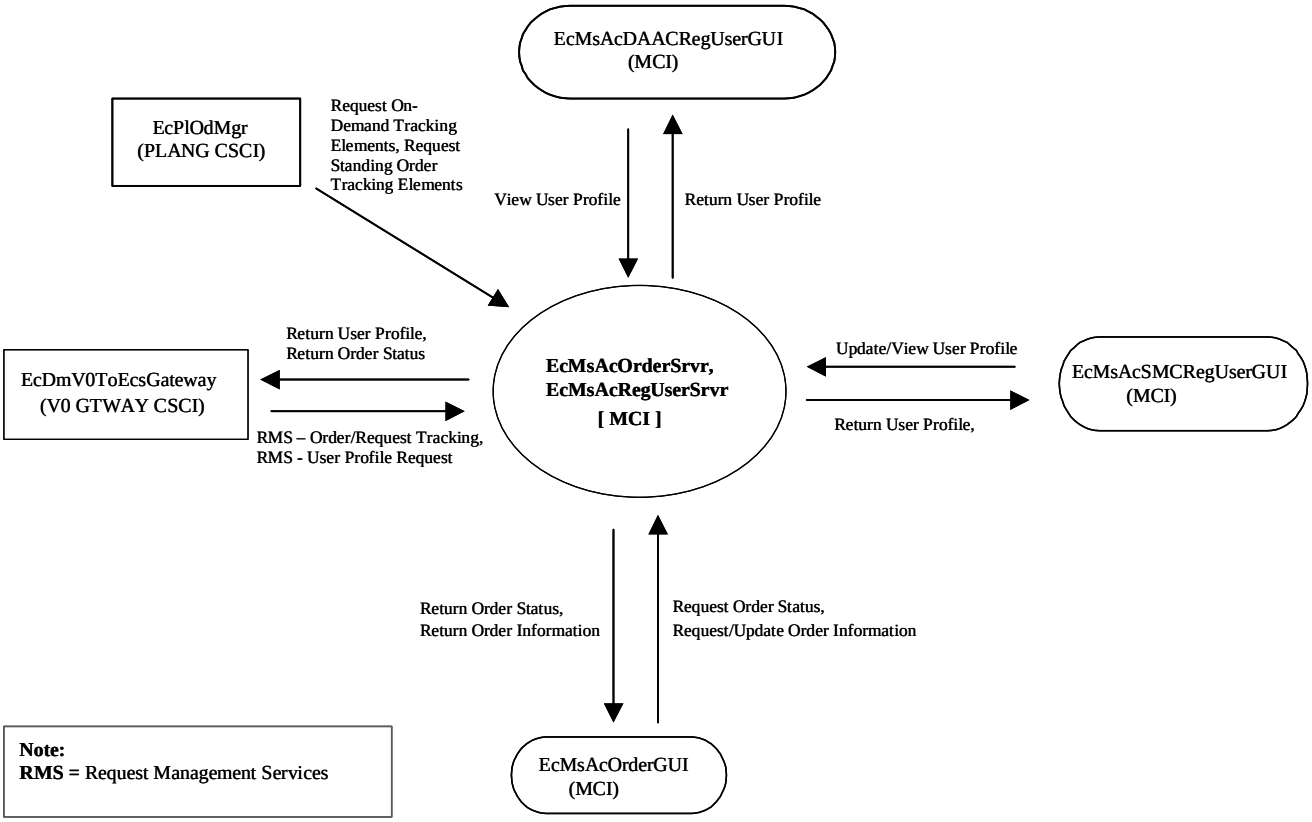


Figure 4.9-14. Management Software CSCI (MCI) Architecture Diagram

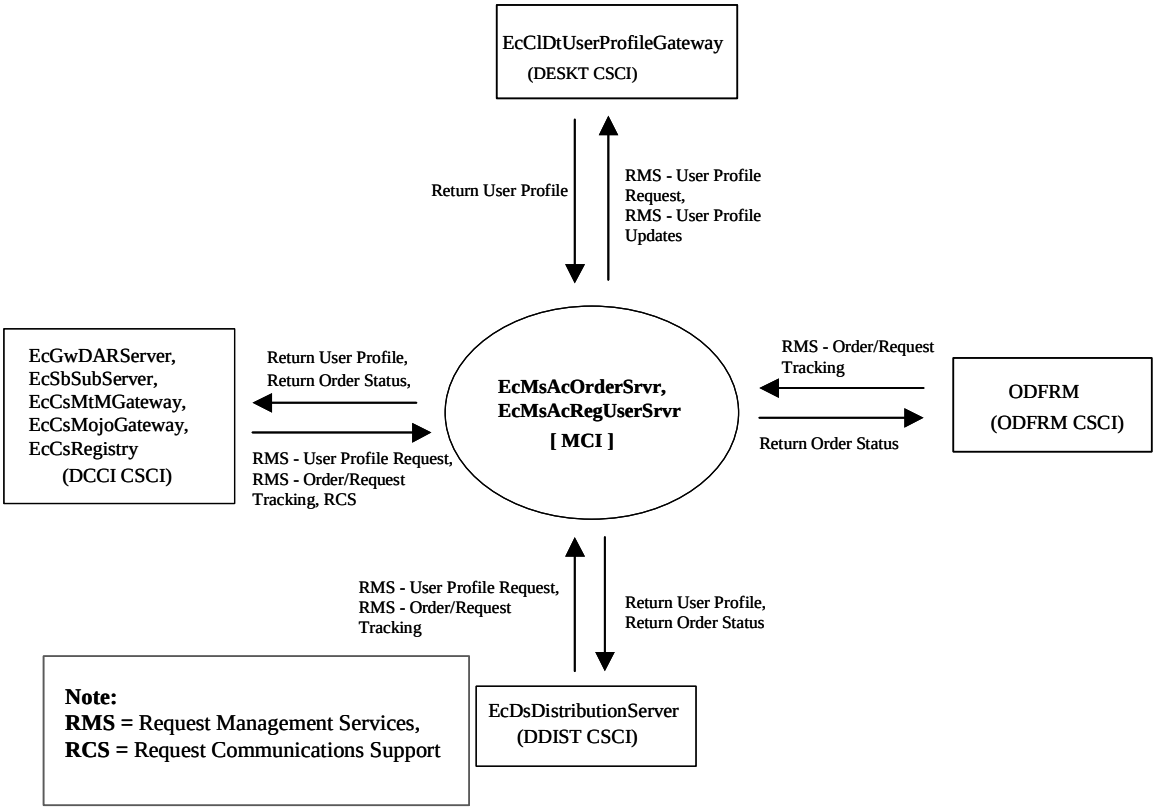


Figure 4.9-14. Management Software CSCI (MCI) Architecture Diagram (cont.)

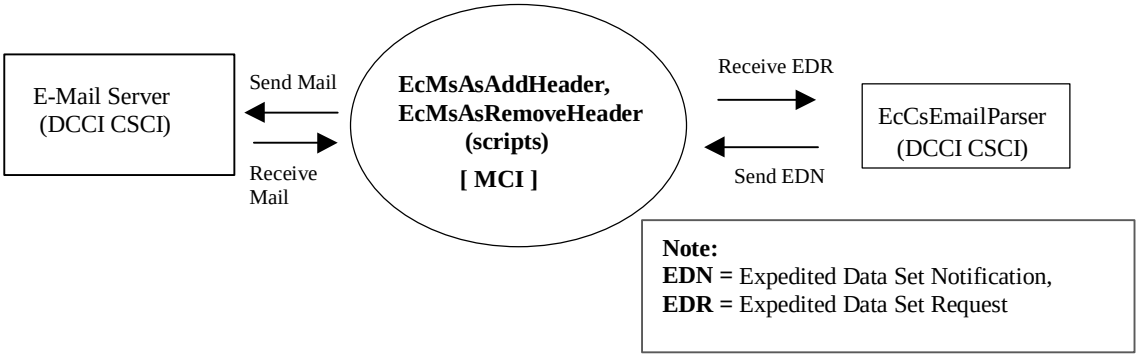


Figure 4.9-14. Management Software CSCI (MCI) Architecture Diagram (cont.)

4.9.1.4 Management Software Process Descriptions

Table 4.9-26 provides descriptions of the Management Software CSCI (MCI) processes shown in the MCI architecture diagram.

Table 4.9-26. Management Software CSCI (MCI) Processes (1 of 2)

Process	Type	COTS/ Developed	Functionality
EcMsAcRegUserSrvr	Server	Developed	The User Registration Server provides an internal interface to the User Registration GUI and an external interface to other CSCIs/CSCs. The functions are: 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 The EcMsAcRegUserSrvr supports: • Single requests at a time • Multiple concurrent requests • Asynchronous request processing • Multiple threads within a single request
EcMsAcSMCRegUserGUI	GUI	Developed	The User Registration graphical user interface enables the viewing and updating of user profiles. The GUI enables the user to: 1. Delete an ECS user 2. Modify an ECS user profile 3. Change the V0 gateway password 4. Change ASTER category and send e-mail 5. Change the DAR privilege 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.
EcMsAcDAACRegUserGUI	GUI	Developed	The User Registration graphical user interface enables the viewing of user profiles. The GUI enables the user to list and view ECS user profiles.

Table 4.9-26. Management Software CSCI (MCI) Processes (2 of 2)

Process	Type	COTS/ Developed	Functionality
EcMsAcOrderSrvr	Server	Developed	The Order Tracking Server provides an external interface to other CSCIs/CSCs. The functions are: 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: • Single requests at a time • Multiple concurrent requests • Asynchronous request processing • Multiple threads within a single request
EcMsAsAddHeader	Script	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	Script	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

Table 4.9-27 provides descriptions of the interface events shown in the Management Software CSCI (MCI) architecture diagram.

Table 4.9-27. Management Software CSCI (MCI) Process Interface Events (1 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
Request Management Services (RMS)	One per service request	N/A	N/A	The 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 includes the items below.
RMS – (cont.)	At system startup or shutdown and for restarts	*Processes:* EcMsAcOrderSrvr, EcMsAcRegUserSrvr	DAAC unique startup scripts	**System startup and shutdown -** Please refer to the release-related, current version of the Mission Operations Procedures for the ECS Project document (611) and the current ECS Project Training Material document (625), identified in Section 2.2.1 of this document.

Table 4.9-27. Management Software CSCI (MCI) Process Interface Events (2 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
(RMS – cont.)	One per request	*Process:* EcMsAcOrderSrvr *Library:* MsAcClnt *Class:* EcAcOrderCMgr	*Process:* EcDmV0ToEcsGateway *Class:* DmGwRequestReceiver CGI Interface, CGI Scripts	The EcMsAcOrderSrvr interfaces with other processes to perform the following: **Order/Request Tracking** - The **EcDmV0ToEcsGateway** interfaces with the EcMsAcOrderSrvr (Order/Request Tracking service) to create a user product order record. The **ODFRM CSCI**, via the iPlanet Web Server, uses CGI scripts to interface with the MCI Order/Request Tracking service to create a user product order and submit the order to the PLS. In addition, the **MTMGW CSC** can also create a user product order and retrieve it from the EcMsAcOrderSrvr.

Table 4.9-27. Management Software CSCI (MCI) Process Interface Events (3 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
(RMS – cont.)	One profile per request	***Process:*** EcMsAcRegUserSrvr, *Libraries:* MsAcClnt, MsAcComm *Class:* EcAcProfileMgr ***Process:*** EcMsAcOrderSrvr *Library:* MsAcClnt *Class:* EcAcOrderCMgr ***Process:*** EcCsMtMGateway *Libraries:* MsAcClnt, MsAcComm	***CSS Process:*** EcGwDARServer *Class:* EcGwDARGatewayRequest_S ***CSS Process:*** EcSbSubServer ***DMS Process:*** EcDmV0ToEcsGateway *Class:* DmGwRequestReceiver ***CLS Process:*** EcClDtUserProfileGateway *Class:* ClDtProfileServer ***CSS Process:*** EcCsMojoGateway *Library:* EcCsMojoGateway *Class:* EcMjRetrieveProfileProxy ***DSS Process:*** EcDsDistributionServer *Library:* DsDdSSh *Class:* DsDdMedia ***CSS Process:*** EcCsMtMGateway *Class:* EcCsMtMAcctSrvMgr	**User Profile Request** - The EcMsAcRegUserSrvr provides requesting processes (**EcGwDARServer**, **EcSbSubServer**, **EcCsMtMGateway**, **EcDmV0ToEcsGateway**, **EcClDtUserProfileGateway**, **EcCsMojoGateway** and **EcDsDistributionServer**) with user profile information such as e-mail address and shipping address to support their processing activities.
View User Profile	One per view user profile request	*Process:* EcMsAcRegUserSrvr *Libraries:* MsAcClnt, MsAcComm	M&O staff *Process:* EcMsAcDAACRegUserGUI *Libraries:* MsAcClnt, MsAcComm	The M&O staff request, via the **EcMsAcDAACRegUserGUI**, to retrieve a user profile from the EcMsAcRegUserSrvr.

Table 4.9-27. Management Software CSCI (MCI) Process Interface Events (4 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
Return User Profile	One per return of user profile	*Processes:* EcMsAcSMCRegUserGUI, EcMsAcDAACRegUserGUI, EcDmV0ToEcsGateway, EcClDtUserProfileGateway, EcDsDistributionServer, EcGwDARServer, EcSbSubServer, EcCsMtMGateway, EcCsMojoGateway *Libraries:* MsAcClnt, MsAcComm	*Process:* EcMsAcRegUserSrvr *Libraries:* MsAcClnt, MsAcComm	The EcMsAcRegUserSrvr returns user profile information provided from the Sybase ASE to the requester at the **EcMsAcSMCRegUserGUI** or the **EcMsAcDAACRegUserGUI** and to the **EcDmV0ToEcsGateway**, **EcClDtUserProfileGateway, EcDsDistributionServer, EcGwDARServer, EcSbSubServer, EcCsMtMGateway** and the **EcCsMojoGateway**.
Update/View User Profile	One per create/ update user profile	*Process:* EcMsAcRegUserSrvr *Libraries:* MsAcClnt, MsAcComm	M&O staff *Process:* EcMsAcSMCRegUserGUI *Libraries:* MsAcClnt, MsAcComm *Class:* MsAcRegUsrUtl	The M&O staff sends requests to modify or review user profile information via the **EcMsAcSMCRegUserGUI**.
Request Order Status	One per request order status	*Process:* EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	M&O Staff *Process:* EcMsAcOrderGUI *Libraries:* 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.

Table 4.9-27. Management Software CSCI (MCI) Process Interface Events (5 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
Request/Update Order Information	One per request/update order information	*Process:* EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	M&O Staff *Process:* EcMsAcOrderGUI *Libraries:* 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	*Processes:* EcMsAcOrderGUI, EcDmV0ToEcsGateway, ODFRM, EcDsDistributionServer, EcCsMtMGateway *Libraries:* MsAcClnt, MsAcComm	*Process:* EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	The EcMsAcOrderSrvr receives the product order status from the Sybase ASE and returns the status to the **EcMsAcOrderGUI**, the **EcDmV0ToEcsGateway**, **ODFRM, EcDsDistributionServer** and the **EcCsMtMGateway**.
Return Order Information	One per return of order information	*Process:* EcMsAcOrderGUI *Libraries:* MsAcClnt, MsAcComm	*Process:* EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	The EcMsAcOrderSrvr receives the product order information from the Sybase ASE and returns the order information to the **EcMsAcOrderGUI**.
Request On-Demand Tracking Elements	Per order	*Process:* EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	*Process:* EcPlOdMgr *Library:* PlCore2	The **EcPlOdMgr** requests On-demand tracking elements (i.e., Order ID and Request ID) from the EcMsAcOrderSrvr.
Request Standing Order Tracking Elements	Per order	*Process:* EcMsAcOrderSrvr *Libraries:* MsAcClnt, MsAcComm	*Process:* EcPlOdMgr *Library:* PlCore2	The **EcPlOdMgr** requests standing order tracking elements (i.e., Order ID and Request ID) from the EcMsAcOrderSrvr.

Table 4.9-27. Management Software CSCI (MCI) Process Interface Events (6 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
Request Communications Support	Request service(s) as required	***Process:*** EcCsIdNameServer ***Libraries:*** EcPf, Middleware, FoNs, FoIp, oodce *Classes:* EcPfManagedServer, CCSMdwNameServer, FoNsNameServerProxy, CCSMdwRwNetProxy ***Library (Common):*** EcUr *Class:* EcUrServerUR ***Library:*** event *Class:* EcLgErrorMsg ***Process:*** EcCsRegistry *Library:* EcCsRegistry *Class:* EcRgRegistryServer_C	*Processes:* EcMsAcOrderSrvr, EcMsAcRegUserSrvr *Libraries:* MsAcClnt, MsAcComm	The **DCCI CSCI** provides a library of services available to each SDPS and CSMS CSCI/CSC. The CSCI services required to perform specific assignments are requested from the DCCI CSCI. These services include: • CCS Middleware Support • Database Connection Services • Name/Address Services • Password Services • Server Request Framework (SRF) • Universal Reference (UR) • Error/Event Logging • Fault Handling Services • Mode Information • Query Registry - Retrieving the requested configuration attribute-value pairs from the Configuration Registry

Table 4.9-27. Management Software CSCI (MCI) Process Interface Events (7 of 7)

Event	Event Frequency	Interface	Initiated By	Event Description
Receive EDR	One per EDR send	*Process:* EcCsEmailParser	*Script:* 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	*Script:* EcMsAsAddHeader	*Process:* EcCsEmailParser *Class:* EcCsEmailParser	The **EcCsEmailParser** sends the Expedited Data Set Notification (EDN) information to the EcMsAsAddHeader script in the MCI to have a header added.
Send Mail	Per user request	*Library:* libc API: System ( )	*Script:* EcMsAcAddHeader.pl *Library:* CsEmMailRelA *Class:* CsEmMailRelA	The EcCsEmailParser sends an e-mail message to the EcMsAcAddHeader.pl script to add a header and invokes a process to send E-mail via an API to the **E-Mail Server**.
Receive Mail	Per user request	*Scripts:* EcMsAcRemoveHeader.pl, EcMsAcAddHeader.pl *Library:* CsEmMailRelA *Class:* CsEmMailRelA	*Library:* libc API: System ( )	The EcMsAcRemoveHeader.pl retrieves mail from the **E-mail Server**, removes the e-mail message header and sends the mail to the EcCsEmailParser CSC for distribution to the appropriate user.

4.9.1.6 Management Software Data Stores

Data stores are not applicable for the Management Software CSCI.

4.9.2 Management Logistics Computer Software Configuration Item Description

4.9.2.1 Management Logistics Functional Overview

The Management Logistics CSCI (MLCI) supports the configuration management of the ECS. The MLCI is the following three CSCs:

- Inventory, Logistics, Maintenance (ILM) Manager: The ILM CSC maintains property records about contract purchased items, storing information such as vendor, date of receipt, installation, and warranty expiration. The ILM CSC also maintains maintenance

records about contract purchased items as well as license management records about COTS software.

- Software Change Manager: The Software Change Manager CSC supports maintenance and change control of the science software configuration at each DAAC.
- Software License Manager: The Software License Manager CSC monitors and controls licensing of COTS products installed in the ECS.

4.9.2.1.1 MLCI - Inventory/Logistics/Maintenance Manager Computer Software Component Description

4.9.2.1.1.1 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 items including hardware, COTS software and associated licenses, 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 about hardware -- including corrective maintenance.

The ILM Manager gives authorized users access to property, license and maintenance data via a graphical user interface (GUI). The user can query for and display an individual record or a list of a set of records that meets specified search criteria. Reporting capabilities are available to the user by accessing the report menu and entering valid values. The user can also obtain ad-hoc reports of ILM data and transmit the reports to the screen, a file, or a printer.

- The functionality for the Inventory/Logistics/Maintenance Manager is implemented via the Remedy Action Request System and ILM related custom scripts.

4.9.2.1.1.2 Inventory/Logistics/Maintenance Manager Context

The ILM Manager does not have an interface with any other subsystem, CSCIs or CSCs.

4.9.2.1.1.3 Inventory/Logistics/Maintenance Manager Architecture

The ILM Manager is implemented as a Remedy ARS application. Remedy ARS is a client/server COTS application that facilitates the ILM function through a set of inventory, logistics, license and maintenance transactions supporting forms and processing capabilities. Remedy ILM's data is stored in a Sybase relational database. The database is shared with the Trouble Ticket CSC. Data records can be exported via formatted data files for use by the ILM Manager applications at other sites. Figure 4.9-15 is the ILM Manager architecture diagram.

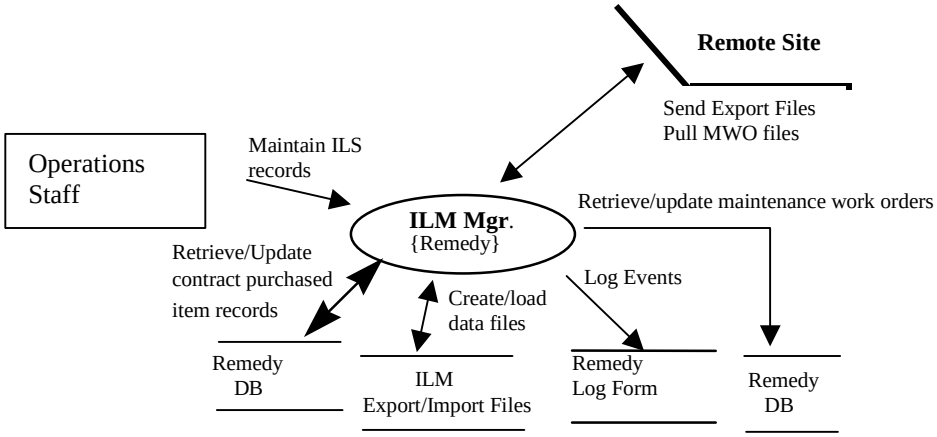


Figure 4.9-15. Inventory/Logistics/Maintenance (ILM) Manager Architecture Diagram

4.9.2.1.1.4 Inventory/Logistics/Maintenance Manager Process Descriptions

The Remedy ILM application is initiated by the Remedy PC User tool. Once logged into the user tool, the user has access to a variety of forms/capabilities that he or she is authorized to use. These forms/capabilities enables the user to perform the functions summarized in Table 4.9-28.

Table 4.9-28. Inventory/Logistics/Maintenance (ILM) Manager Processes

Process	Type	COTS / Developed	Functionality
ILM Mgr. (Remedy ARS)	Other	COTS	The ILM performs the following tasks: • Captures and maintains all pertinent data for project hardware and COTS software licenses • Manages, distributes and reports on consumables and spares • Manages, controls and reports on 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, manufacturer, and internal usage codes

4.9.2.1.1.5 Inventory/Logistics/Maintenance Manager Process Interfaces

Table 4.9-29 provides descriptions of the interface events shown in the ILM Manager architecture diagram.

Table 4.9-29. ILM Manager Process Interface Events (1 of 2)

Event	Event Frequency	Interface	Initiated By	Event Description
Send Export Files	One per data export request	*Data Store:* ILM Import/Export Files	*Program:* CM File Export Script	Export script executes at a preset time and causes the transfer of a tar file containing exported ILM records to one or more **remote sites**, via the FTP service.
Retrieve/update maintenance work orders	Once per retrieval or update; Daily for DAACs	*Data Store:* Remedy ARS DB	*Program:* ILM Mgr. (Remedy ILM)	Maintains and retrieves information for maintenance work orders in the **Remedy ARS DB**.
Log events	Once per update	*Data Store:* Remedy ILM Transaction Log Form	*Program:* ILM Mgr. (Remedy ILM)	Logs information on activities performed by this COTS package in the **Remedy ILM Transaction Log Form**.
Create/Load data files	Once per creation or retrieval; Daily for DAACs	*Data Store:* ILM Export/Import Files	*Program:* Remedy ILM scripts	Maintains data files of contract purchased equipment from other sites (DAACs, EOC, and SMC) in **ILM Export/Import Files** and sends information to other sites for spare and consumable parts information as well as for backup/retrieve purposes.
Retrieve/Update contract purchased item records	Once per retrieval or update	*Data Store:* Remedy ILM DB	*Program:* ILM Mgr. (Remedy ILM)	Maintains and retrieves contract purchased item information for ECS hardware, COTS software, COTS documentation, spare parts, consumable items such as media, tape, cables and GFE. Tracks warranties and maintenance for equipment in the **Remedy ARS DB**. Tracks license entitlements, licenses, and license allocations for COTS software.

Table 4.9-29. ILM Manager Process Interface Events (2 of 2)

Event	Event Frequency	Interface	Initiated By	Event Description
Maintain ILS records	Once per request to maintain contract purchased equipment or maintenance work order records	*Program:* ILM Manager (Remedy ILM)	Operations Staff *Program:* ILM Manager (Remedy ILM) GUI	Enables the **Operations Staff** to: • • 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

4.9.2.1.1.6 Inventory/Logistics/Maintenance Manager Data Stores

ILM Manager's principal data stores are the XRP-II database and formatted data files used for exporting and importing ILM records. The data store descriptions are provided in Table 4.9-30.

Table 4.9-30. Inventory/Logistics/Maintenance (ILM) Manager Data Stores

Data Store	Type	Functionality
Remedy ILM DB	Database	A non-replicated collection of inventory and maintenance-related data that exists at each site. For ILM, it contains records identifying and describing: • Inventory items • Corrective maintenance • Location data • License management data • Maintenance support information
ILM Export/Import Files	Tar file	Formatted data files created as necessary to exchange the ILM Manager records among sites. Each file contains: • All changed records pertaining to receipts, installations, archives, transfers, relocations, shipments, manual changes, or maintenance that is performed.
Remedy ILM Transaction Log Form	Form	• A collection of transaction records that describe the receipt, installation, relocation, and archiving of inventory items.

4.9.2.1.2 MLCI - Software Change Manager Computer Software Component Description

4.9.2.1.2.1 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.

4.9.2.1.2.2 Software Change Manager Context Diagram

The Software Change Manager does not interact with any CSCIs or CSCs.

4.9.2.1.2.3 Software Change Manager Architecture

The Software Change Manager (ClearCase) does not interface with any external processes.

4.9.2.1.2.4 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 4.9-31 provides a summary of its functions.

Table 4.9-31. Software Change Manager Processes

Process	Type	COTS / Developed	Functionality
SW Change Mgr. (ClearCase)	Program	COTS with custom developed scripts	• 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)

4.9.2.1.2.5 Software Change Manager Process Interface Descriptions

Software Change Manager Process Interface Descriptions are not applicable to the MLCI.

4.9.2.1.2.6 Software Change Manager Data Stores

The Software Change Manager's COTS package, ClearCase's data stores consist of a database and log files. Table 4.9-32 provides descriptions of the data stores shown in the Software Change Manager architecture diagram.

Table 4.9-32. 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	File	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.

4.9.2.1.3 MLCI - Software License Manager Computer Software Component Description

4.9.2.1.3.1 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.

4.9.2.1.3.2 Software License Manager Context

Software License Manager runs at every ECS site, providing local network licensing services to requesting COTS applications and ECS operators. The Software License Manager does not interact with any other CSCIs or CSCs.

4.9.2.1.3.3 Software License Manager Architecture

Figure 4.9-16 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. A redundant server implemenation is currently used at all sites for both the Sun and SGI hosts.

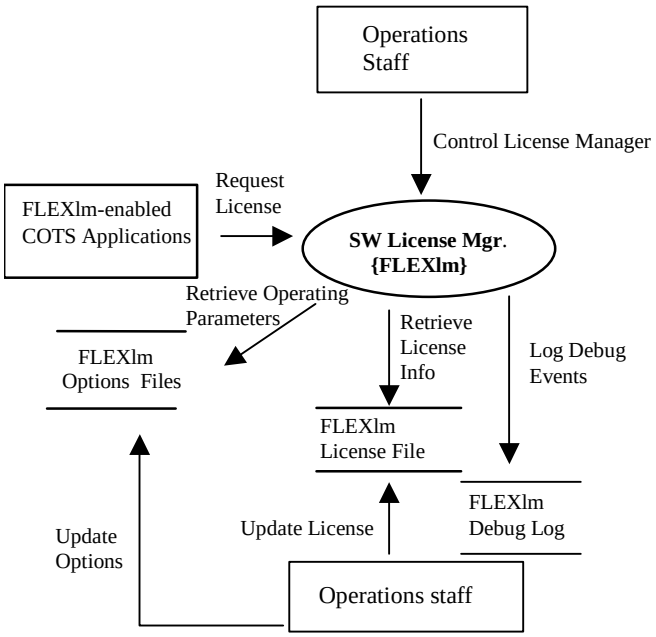


Figure 4.9-16. Software License Manager Architecture Diagram

4.9.2.1.3.4 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 4.9-33 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 4.9-33 provides a description of the process involved in software licenses management for a local site.

Table 4.9-33. Software License Manager Processes

Process	Type	COTS / Developed	Functionality
Software License Mgr. (FLEXlm)	Program	COTS	The FLEXlm server daemon (lmgrd) with its associated command line utilities: • 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	Program	COTS	Client software within vendor products communicates with FLEXlm's license server and vendor daemons to request licenses for product users to run.

4.9.2.1.3.5 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 4.9-34 provides descriptions of the interface events shown in the Software License Manager architecture diagram.

Table 4.9-34. Software License Manager Process Interface Events (1 of 2)

Event	Event Frequency	Interface	Initiated By	Event Description
Control License Manager	One per event	Process: SW License Mgr. (COTS) (FLEXlm)	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	*Data Store:* FLEXlm Debug Log	*Process:* SW License Mgr. (FLEXlm) (COTS)	The FLEXlm 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) in the **FLEXlm Debug Log**.

Table 4.9-34. Software License Manager Process Interface Events (2 of 2)

Event	Event Frequency	Interface	Initiated By	Event Description
Update License	One per update request	*Data Store:* *FLEXlm* License File	Operations Staff	The **Operations Staff** adds and removes the managed FLEXlm software licenses.
Retrieve License Info	One per request	*Data Store:* FLEXlm License File	*Process:* SW License Mgr. (FLEXlm) (COTS)	Reads license provisions for one or more FLEXlm-enabled COTS products from the **FLEXlm License File**.
Update Options	One per parameter change	*Data Store:* FLEXlm Options Files	Operations Staff *Process:* Unix file editor	Store parameters for regulating license usage and event logging in the **FLEXlm Options Files**.
Retrieve Operating Parameters	One per license request	SW License Mgr.(COTS) (FLEXlm)	*Process:* SW License Mgr. (FLEXlm) (COTS)	Retrieve parameters for regulating license usage and event logging from the **FLEXlm Options Files**.
Request License	One per license request	*Process:* SW License Mgr. (COTS) (FLEXlm)	*Programs:* Flexlm - enabled COTS applications	Communication among license servers and clients to establish connections and checkout, check-in, and monitor activity of licenses are obtained from **FLEXlm-enabled COTS applications**.

4.9.2.1.3.6 Software License Manager Data Stores

License Manager's principal data stores are the FLEXlm license, debug, and option files. FLEXlm files are described in the FLEXlm End User Manual. Table 4.9-35 provides descriptions of the data stores shown in the Software License Manager architecture diagram.

Table 4.9-35. Software License Manager Data Stores

Data Store	Type	Functionality
FLEXlm license file	Text file	A collection of records containing license provisions and passwords for one or more FLEXlm-enabled COTS products. They identify: • 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 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 FLEXlm End User Manual.
FLEXlm debug log	Text file	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.
FLEXlm options files	Text file	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.

4.9.2.2 Management Logistics Context

The Management Logistics CSCI (MLCI) does not interact with any other CSCIs or CSCs.

4.9.2.3 Management Logistics Architecture

An architecture diagram does not apply to the MLCI because it consists of standalone tools.

4.9.2.4 Management Logistics Process Description

Process descriptions are not applicable for the MLCI.

4.9.2.5 Management Logistics Process Interface Descriptions

Process Interface Descriptions are not applicable for the MLCI.

4.9.2.6 Management Logistics Data Stores

Data Stores are not applicable for the MLCI.

4.9.3 ECS Assistant Script Library

4.9.3.1 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.

4.9.3.2 ECS Assistant Script Library Context

The context diagram is not applicable to the ECS Assistant Script Library.

4.9.3.3 ECS Assistant Script Library Architecture

The architecture diagram is not applicable to the ECS Assistant Script Library.

4.9.3.4 ECS Assistant Script Library Processes

The Process Table is not applicable to the ECS Assistant Script Library.

4.9.3.5 ECS Assistant Script Library Process Interface Descriptions

The ECS Assistant Script Library provides functions available to calling scripts. Table 4.9-36 describes the interface.

Usage: /ecs/formal/COMMON/scripts/EcCoScriptlib command [args]

Table 4.9-36. ECS Assistant Script Functions (1 of 3)

Function	Functionality
add_cds_entry	Add CDS entry.
add_client	Add CCS Middleware client.
add_size	Record the package size in the .install_stats file
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.

Table 4.9-36. ECS Assistant Script Functions (2 of 3)

Function	Functionality
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.
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 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.
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.

Table 4.9-36. ECS Assistant Script Functions (3 of 3)

Function	Functionality
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.
package_size	Return the size of the contents of the package in kilobytes.
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.
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.

4.9.3.6 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 4.9-37. The file maps application names to application ids.

Table 4.9-37. 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 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 4.9-38 (whether it is delivered or not), there is a tag and the parameter value pairs, which describe the is the .cfgparms file format.

Table 4.9-38. 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 6B.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 4.9-39.

Table 4.9-39. Cfgpatch file format

Define a macro									
Action	**Parameter**		**Value**						
DEFINE	REVISION_LEVEL		6B.05.EPSILON.01						
ALL files									
Action	**Site**	**Host**	**Mode**	**Subsys**	**Comp**	**Filename**	**Tag**	**Parameter**	**ValueList**
ADD	ALL	ALL	ALL	ALL	ALL	.dbparms	Rev Level	Revision Level	%${REVISION_LEVEL}
ADD	ALL	ALL	ALL	ALL	ALL	.cfgparms	Rev Level	Revision Level	%${REVISION_LEVEL}
GSFC .cfgparms changes									
Action	**Site**	**Host**	**Mode**	**Subsys**	**Comp**	**Filename**	**Tag**	**Paramet**	**ValueList**
ADD	GSFC	ALL	ALL	CLS	EcCl	.cfgparms	EcClWbJdtMkcfg	SUBSCRIPTION_ESDT_SN	AST_L1BT
ADD	GSFC	ALL	ALL	CLS	EcCl	.cfgparms	EcClWbJdtMkcfg	SUBSCRIPTION_ESDT_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 4.9-40 is a sample of the file format.

Table 4.9-40. Components file format

Mnemonic	Directory	CSCI
EcDsSr	SDSRV	Science Data Server
EcDsSt	STMGT	Storage Management
EcDsDd	Distribution	Data Distribution

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, and *sgi6n32*, which applies to SGI hosts. The other entries are in the form of parameter value pairs. The resulting file consists 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 4.9-41 is a sample of the file format.

Table 4.9-41. Executables file format

Program	IU Name	Type	ProgID	Application Name
EcClOdProductRequest	EcClOdAsterOnDemand.iu	C	1000005	EcClOdProductRequestApp
EcClDtDesktopDaacUser	EcClDtDesktopDaacUser.iu	G	1000100	EcClDtDesktopApp
EcClDtDesktopSciUser	EcClDtDesktopSciUser.iu	G	NONE	EcClDtDesktopApp
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 4.9-42 is a sample of the file format.

Table 4.9-42. 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 4.9-43. 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 4.9-43. 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	
${MODE_STAGING_DIR}/CUSTOM/bin/${SUBSYS_DIR_MODE}	
${CMTOP/${SUBSYS_DIR_CM}/install/CUSTOM/bin/${SUBSYS_DIR_MODE}	
${CMTOP}/${STAGE}/${SUBSYS_DIR_CM}/bin/${ARCH}	
${BINDIR}	

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 *SGI or SUN* separated by pipe symbols "|". The EcCoMkDeliver script uses the packages file to determine which packages need to be delivered when making the tar files for a drop on a particular architecture. Table 4.9-44 is a sample of the file format.

Table 4.9-44. Packages file format

Package	Subsys	Comp	Platforms	Comment
.EcClEOSView.pkg	CLS	EcCl	SUN	
.EcClINTFCSVR.pkg	CLS	EcCl	SUN	
.EcClINTFCSVR_EDC.pkg	CLS	EcCl	SUN	
.EcClJdt.pkg	CLS	EcCl	SUN	
.EcClOdAsterOnDemand.pkg	CLS	EcCl	SUN	
EcCsCommon.pkg	CSS	EcCs	SGI | SUN	
EcCsDarMainMSSSVR_EDC.pkg	CSS	EcCs	SUN	
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 solves this. The format is shown in table 4.9-45.

Table 4.9-45. Sitehostmap file format (1 of 3)

Symbolic Name	EDC	GSFC	LaRC	MDAAC	MSMC	NSIDC	SMC	VATC	VSMC	EDFFRT
AsterDemWs	e0ass03									

Table 4.9-45. Sitehostmap file format (2 of 3)

Symbolic Name	EDC	GSFC	LaRC	MDAAC	MSMC	NSIDC	SMC	VATC	VSMC	EDFFRT
AsterLutDb01	e0asss01									
AsterLutDb02	e0asss02									
BbsServer					uhoh		m0csss02		t1sms12	
CssServer	e0csss02	g0csss02	10csss02	onfire	chammy	n0csss02	m0csss03	t1csss01	t1sms09	
FtpServer01					monday		m0csss05		t1sms11	

Table 4.9-45. Sitehostmap file format (3 of 3)

Symbolic Name	EDC	GSFC	LaRC	MDAAC	MSMC	NSIDC	SMC	VATC	VSMC	EDFFRT
FtpServer02							m0css04			
DbaOperationWs	e0dmh02	g0dmh01	10dmh02	darkwind		n0dmh02		t1dmh03		

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 4.9-46.

Table 4.9-46. Sitemap file format (1 of 2)

Entry	Package(s)
EDC:e0ais02:CLS:EcCl # AIT Workstation (Default)	EcClEOSView.pkg
EDC:e0ais03:CLS:EcCl # AIT Workstation/DBMS Server	EcClEOSView.pkg
EDC:e0ins01:CLS:EcCl # Interface Server 02 # Principal for CSS, Backup for DMS, CLS Servers	.EcCsINTFCSVR.pkg .EcCsINTFCSVR_EDC.pkg .EcClJdt.pkg .EcClOdAsterOnDemand.pkg
EDC:e0ins02:CLS:EcCl # Interface Server 01 # Principal for DMS, CLS Servers, Backup for CSS	.EcCsINTFCSVR.pkg .EcCsINTFCSVR_EDC.pkg .EcClJdt.pkg .EcClOdAsterOnDemand.pkg

Table 4.9-46. Sitemap file format (2 of 2)

Entry	Package(s)
EDC:e0pls03:CLS:EcCl # Planning/Management WS 01 (Default)	EcClEOSView.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 4.9-47.

Table 4.9-47. Subsystems file format

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
om	OMS	OMS	OMS	Order Manager
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 4.9-48.

Table 4.9-48. 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 4.9-49.

Table 4.9-49. Start Scripts (1 of 2)

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.

Table 4.9-49. Start Scripts (2 of 2)

Start Script	Functionality
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 Mkcfg Scripts

The ECS Mkcfg scripts call the ECS Assistant Script Library for a variety of services, which are shown in the Table 4.9-50.

Table 4.9-50. Mkcfg Scripts

Mkcfg 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.

4.9.4 Systems Management Subsystem Hardware Components

4.9.4.1 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.

Document 920-TDx-001 provides descriptions of the System Management HWCI.

The MSS File Server and CM Server are configured similarly with additional RAM allocated to the File Server due to file distribution loading. One MSS software CSCI, MLCI, runs on these hosts. Some of the key MLCI functions are the Software Change Manager (ClearCase) andSoftware Licensing Manager (FLEXlm). 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.

The MSS Server and MSS Server Backup are High End Workstation class machines. One MSS software CSCI, MCI, runs on these hosts. Some of the key MCI functions are network, enterprise, fault and performance management. These functions are supported by WhatsUp Gold. An additional key MCI component is trouble ticket (Remedy). Detailed mappings can be found per the site-specific hardware/software mapping, baseline document number 920-TDx-002.

A Sun RAID device is dual ported between both hosts and provides storage for Remedy data. A detailed configuration is specified per site-specific disk partition, baseline document number 922-TDx-031.

The Tape Backup Server is a SUN Server class machine. Detailed 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 DLTL (Digital Linear Tape Library) used for global system DAAC backups. One MSS software component runs on this host and is the MCI. The key MCI functions are the network backup and restore components (Legato Networker). Detailed mappings can be found per the site-specific hardware/software mapping, baseline document number 920-TDx-002.

A DLTL with more then 1 terabyte (TB) of capacity is directly attached to the Tape Backup Server. Via the Legato Networker Server, system data copies and restores are performed with the Tape Backup Server functioning as the intermediate between the Legato clients and the DLT. A detailed 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. Detailed 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, 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, 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, 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 protocol is 10/100/1000 Mbps Ethernet.

High-speed Gig Ethernet networks form part of the networks at the DAACs to handle the high data volumes between the Processing and Data Server subsystems.

Internetworking Hardware HWCI (INCI)

This HWCI provides the networking hardware for internal and external DAAC and SMC connectivity. The HWCI includes Ethernet switches and cabling; routers 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. The topology consists of a Production Network, and a High-speed Gig Ethernet Network. A Portus Firewall protects the Production network. The introduction of the High-speed Gig Ethernet 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. For instance LaRC has a direct connection to NASA Integrated Services Network (NISN), and GSFC and EDC no longer have an ECS router.

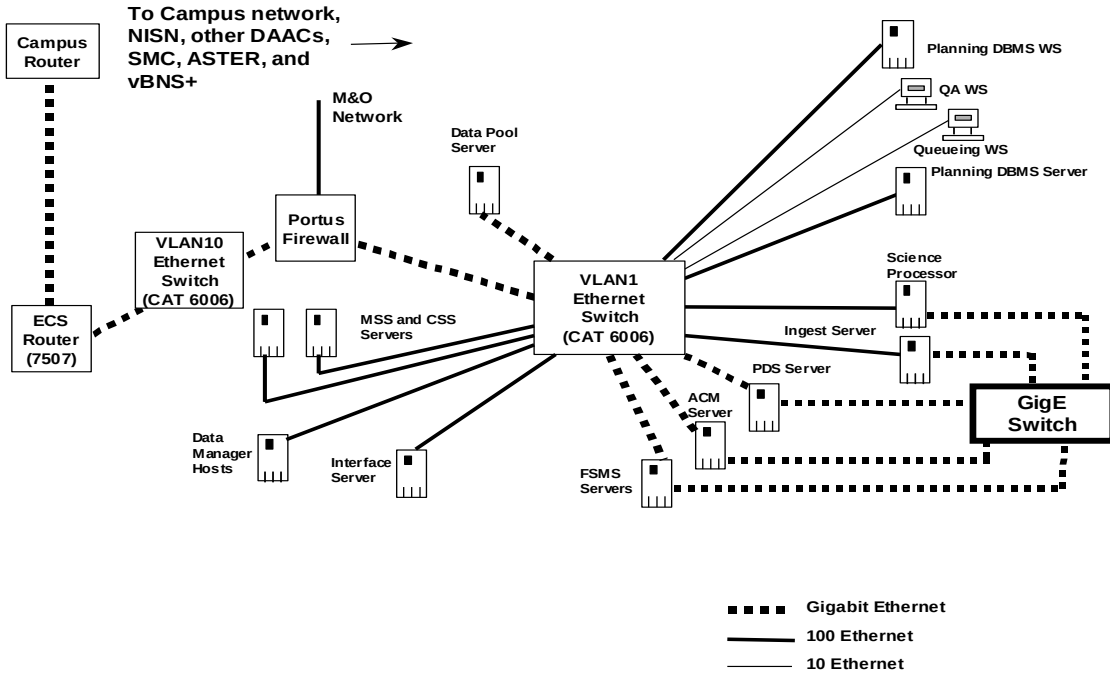


Figure 4.10-1. DAAC Networks: Generic Architecture Diagram

The Production Network consists of a Catalyst 6006 multi-port Ethernet Switch. All servers, workstations, printers, PCs and x-terms are connected to individual switch ports.

The High-speed Gig Ethernet Network interconnects Data Server hosts/devices, Ingest Processors, PDS Processor and Science Processors in order to provide a high-speed network to handle the large data transfers between the multiple subsystems. The High-speed Gig Ethernet network is implemented via a central Gig Ethernet switch, connected directly to the high-powered processing and storage hosts. The High-speed Gig Ethernet 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 4.10-2, consists of a Catalyst 2912 Ethernet Switch. Each server, workstation, printer, and x-term is connected to individual switch ports. A Portus Firewall protects the SMC network.

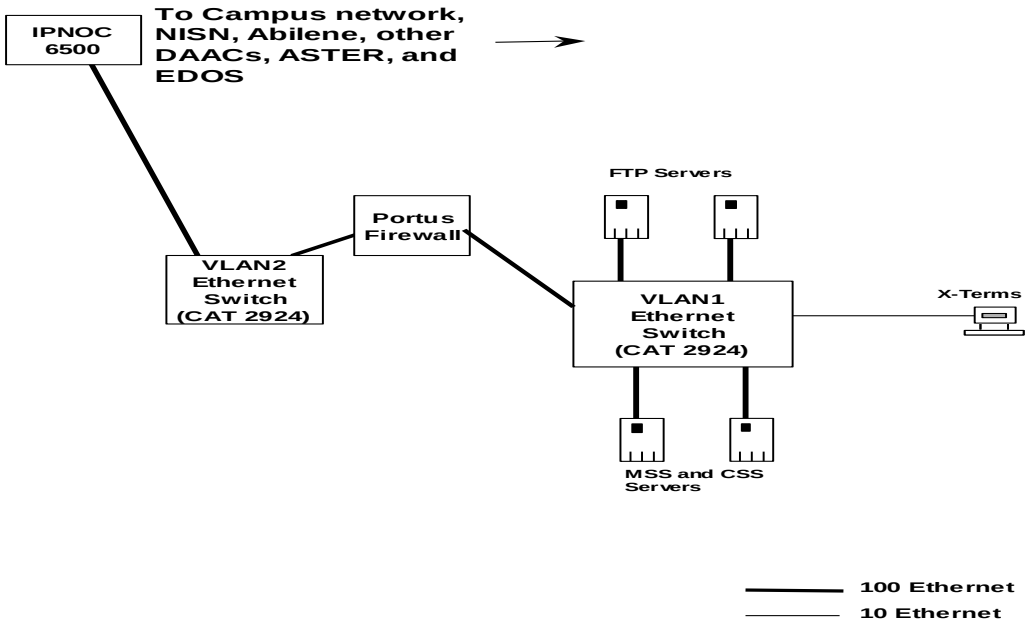


Figure 4.10-2. SMC Network Architecture Diagram

4.10.1.3 DAAC Addressing and Routing Architecture

The Planning and Data Processing, Storage Area Network (SAN), Ingest, MSS, CSS, and Data Server/Data Management subsystems (collectively known as the Production Network) are connected to the Ethernet switch. They are assigned a Class C address space. A subset of hosts (belonging to the Data Server, Ingest and Data Processing subsystems) also has interfaces connected to a High-speed Gig Ethernet switch forming a High-speed production network. The

hosts are assigned private addresses as specified in RFC 1918 (as of 02/96). 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 High-speed Gig Ethernet networks) is provided by EMSnet from Class C address blocks designated by NISN.

Routing Information Protocol (RIP) is the main protocol used to route IP packets within ECS. Static, Open Shortest Path First (OSPF), and Border Gateway Protocol (BGP) protocols are also used to route IP packets to/from external networks. ECS Production Networks are advertised to all ECS via EMSnet.

4.10.1.4 Network-based Security Architecture

The network architecture provides a strong level of security by implementation a Proxy Firewall (Portus). This firewall blocks incoming network traffic unless there is a rule specifically allowing the traffic to pass into the DAACs and SMC. Note that in addition to network-based security; ECS has implemented other security measures, such as secure shell (SSH) and host access lists (ACLs), which are discussed in the CSS sections of this document.

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. 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 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. 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

4.10.2 Network COTS Hardware

The DAAC and SMC LANs contain three types of COTS hardware: Firewall, Ethernet switches, and Routers. All hosts in the DAACs and SMC are attached to Ethernet switches. The Routers are used to provide access to EMSnet and external networks (NISN, Abilene, and Campus nets). The High-speed Gig Ethernet switches connect the Data Server and processing hosts with a high-speed fabric to be used for transferring large volumes of data between the multiple subsystems (Data Server, Ingest, PDS, and Data Processing). Table 4.10-2 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.

Table 4.10-2. Networking Hardware for ECS Networks

Networking Hardware	Vendor
Firewall	Portus IBM PowerPC Server
Router (ECS Router)	Cisco 7507
Gig Ethernet Switch	Catalyst 3550
Ethernet Switch	Catalyst 6006
Ethernet Switch	Catalyst 2924
Ethernet Cables	10baseT, 100baseT, or 1000baseT connection to servers, workstations, printers, PCs, and x-terms
Fiber Cable	1000baseSX connection to host

4.10.2.1 ECS Ethernet Switch

The ECS Ethernet switch is the Cisco Catalyst 6006 with multiple 10/100/1000 Mbps ports and powerful packet engines. The switch has a switching fabric of 32Mbps. It forms the core of the ECS Production network by interconnecting all servers, workstations, printers, PCs, and x-terms. The switch has redundant power supply and fan units. It also has redundant packet engines. All modules are hot swappable.

4.10.2.2 ECS Router

The ECS Router is a Cisco 7500 series router running Cisco's Internetwork Operating System (IOS). All routers have Versatile Interface Processor (VIP) boards populated with 100 Mbps Ethernet ports, and 1000 Mbps Ethernet ports. The ECS Router is only used at LaRC and NSIDC andit provides connectivity to EMSnet, and the Internet via its interface with NISN and Abilene.

The ECS Router has redundant power supply and fan units. All interface modules are hot swappable.

At GSFC and EDC, the ECS firewall interfaces directly with the Campus routers which provide all external network connectivity.

4.10.2.3 High-speed Gig Ethernet Switch

The ECS High-speed Gig Ethernet switch is a Cisco Catalyst 3550 switch capable of supporting up to 12 ports. The Gig Ethernet switch forms the core of the High-speed Gig Ethernet fabric interconnecting multiple subsystems providing capacity of up to 1000 Mbps per connection. The ECS High-speed Gig Ethernet switch is part of ECS DAAC networks at all of the DAACs.

4.10.2.4 Firewall

The ECS Firewall is an IBM PowerPC Server. It is a Proxy type firewall, which is capable of supporting several 100/1000 Mbps Ethernet interfaces. 1000 Mbps interfaces are used for the Production network. At the GSFC DAAC, the 1000 Mbps interface between MODIS and the DAAC is protected by the firewall. All Production and M&O networks are connected to the firewall. In addition, the SMC is connected to the firewall.

At GSFC and EDC, the firewall interfaces directly with the Campus routers which provide all external network connectivity.

Document 920-TDx-001 provides descriptions of the Firewall

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) Client-Server 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) via the client/server architecture. A brief description of each of the key concepts for General Process Failure Recovery follows.

4.11.1 Client-Server Rebinding

ECS uses a socket-based infrastructure to provide an rpc-like capability for a distributed object environment. In the infrastructure, the client applications call proxy objects that represent a server's server objects. A proxy object uses the socket name service to find its server object by its known name. The socket name service returns an 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 socket name server is not running. The ECS socket-based infrastructure 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 is it 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.

Client-server 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 Request Manager to request the allocation of staging disk space from a Staging Disk Server. If the Request Manager 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 Request Manager 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 could have paused briefly, but would have been resumed automatically, without a need to restart DDIST.

The socket name service itself is implemented as a server process, as well. However, its TCP/IP address is included as an environment variable in the ECS standard Unix shell and thus known to all applications. Servers send their binding handle(s) to it when they start up and remove them when they shut down. The name service caches the bindings in memory and stores them persistently in a database for recovery in case of a name service crash. Operators need to monitor the name service on a frequent basis so they can restart it immediately if it should fail.

4.11.2 Sybase Reconnecting

A similar approach has been implemented by the ECS infrastructure that provides the interface with the Sybase ASEs. For example, an ECS application may attempt to connect to its Sybase ASE while the 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 (for example, a deadlock error), they release and then re-request the connection. If the cause of the error was a Sybase ASE fault, this connection attempt fails, but causes the infrastructure code to enter the connection re-try loop. If the Sybase ASE is restarted before the retries are exhausted, the application continues normally and now completes the transaction 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. In addition, the MSS order tracking server uses it's own DB connection API 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 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. They are based on the Universal Unique Identifier (UUID), a mechanism for creating system-wide 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, which 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 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, 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 of 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 submitting the request is waiting for a response. Regardless of how the request is handled downstream, whether it succeeded or failed depends on the response 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 if the request does not complete, for example, because the connection is lost to the server to which the request is submitted, 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, the archive insert is retried. 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 acquires interfaces of the E-mail Parser 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 the order 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 it can recover the queue in case of a crash

Instead, a server handling asynchronous requests must keep a queue in a safe place so it can be recovered 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 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 Client-server 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 this 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 logged. They are brought to the attention of the DAAC operations staff for action if there is a corresponding server GUI for the operator (for example, in the case of DDIST). However, not all ECS servers are associated with an operator GUI. In these cases, operators need to monitor the server logs for errors on a regular basis.

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 is 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 left in a device that 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 a file to be removed, and the file does not exist, there is no error, per se, but a warning is generated to caution the client 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 re-submit the request or if it is possible, 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 4.11-1 summarizes the five categories of request responses, and the specific requirements for the application or server currently responsible for the request. ECS servers have been directed to classify their responses accordingly.

Table 4.11-1. 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 a GUI supports the application, 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 a GUI supports the application, 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 socket server in the event 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 automatically attempts to reconnect to the server for a configurable period of time 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 4.11-2.

Table 4.11-2. Fault Handling Policies (1 of 3)

CI	Client Processes	Fault Handling Policy
PLANG, PRONG	EcPlSubMgr	Retry errors: All Subscription processing errors are retried a configurable number of times and for a configurable time period. After this number of times (or time period), the subscription notice is lost. Fatal errors: 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.
	EcPlPREditor_IF EcPlWb	Retry errors: Since these are GUI applications, errors are reported to the user and it is his/her responsibility to retry the request. Fatal errors: Errors are reported to the user.
	EcDpAtStageDAP EcDpAtInsertTestFile EcDpAtInsertStaticFile EcDpAtInsertExeTarFile EcDpAtSSAPGui EcDpAtGetMCF	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. Fatal errors: The User is sent a fatal error message.
	EcDpPrEM	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. 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.
	EcDpPrJobMgmt	If a DPR cannot be assigned to a machine or created in AutoSys, it is left in a PENDING state and the assignment is retried after `DpPrPendingThreadWaitInterval` seconds Fatal errors: N/A
	EcDpPrDeletion	Retry errors: No retries are implemented. Status from DSS is <u>not</u> checked. This can result in missed granule deletions. Fatal errors: N/A
	EcPlOdMgr	Retry errors: Retries errors from the Science Data Server and the Subscription Server. Fatal errors: Logs errors and terminates current on demand requests.

Table 4.11-2. Fault Handling Policies (2 of 3)

CI	Client Processes	Fault Handling Policy
INGST	EcInGran	Retry errors: Errors are retried a configurable number of times, then the granule is suspended. The operator can then either cancel or resume the suspended granule from the Ingest GUI. 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).
	EcInReqMgr	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. Fatal errors: Errors are failed immediately. The external system originating the Ingest request may be notified (depending on the ingest protocol).
	EcInGUI	Retry errors: An error in sending a media ingest request to EcInReqMgr is reported to the operator and the operator can retry. Other retry errors result in the request failing. Fatal errors: Any error results in the request failing.
	EcInPolling	Retry errors: Errors are retried forever, with a delay between retries. Fatal errors: Errors are failed immediately, and are displayed on the Ingest GUI.
	EcInEmailGWServer	Retry errors: N/A Fatal errors: E-mail that cannot be processed is moved to a failed directory, but no operator notification is provided.
SBSRV	EcSbSubServer	Retry errors: Errors are retried for a configured amount of times and suspended. The operators can then either cancel or resume the suspended acquire requests through system provided scripts. Fatal errors: are treated like retry errors.

Table 4.11-2. Fault Handling Policies (3 of 3)

CI	Client Processes	Fault Handling Policy
SDSRV	EcDsScienceDataServer EcDsHdfEosServer	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. Fatal errors: Errors are logged and the request is passed on to the Data Distribution with the appropriate error information. Data Distribution uses this error information in the distribution notification sent to users to inform them of the errors. After a SDSRV fault, the operator restarts the HDF servers manually.
DDIST	EcDsDistributionServer	Errors are presented to the operator via the DDIST GUI. Retry errors: Errors are presented as "Suspended with Errors" and can be resumed by the operator. 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.
STMGT	EcDsStRequestManagerServer EcDsStArchiveServer EcDsStStagingDiskServer EcDsStCacheManagerServer EcDsStPullMonitorServer EcDsStFtpServer EcDsStDTFServer	Retry errors: Errors are passed back to the calling client process. Fatal errors: Errors are passed back to the calling client process.

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 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. Yet another example is the OMS CSCI recognizing a duplicate request originating from the V0ToECSGateway if the gateway is configured not to allow duplicates.

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 in progress result in an exception being thrown back to the client process, which enters a rebinding failure recovery mode (see Client-Server Rebinding section above). Attempts to submit requests while the server is down results 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 also cancels all outstanding requests and reclaims 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 4.11-3. Unless otherwise stated, existing request queues are always retained for warm restarts and cleared for cold starts or cold restarts.

Table 4.11-3. 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 ready are placed into AutoSys from the queue if there are processing slots available.	N/A
	EcPlSubMgr	Any subscriptions that have not been processed are read from a checkpoint file and processed.	N/A
	EcDpPrDeletion	Interim granules marked for deletion are read from the database and are deleted when time out occurs.	N/A
INGST	EcInGran	The EcInGran server automatically restarts submitted requests from the beginning. If a file has been transferred (via FTP), it does not re-do the transfer 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 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 4.11-3. Server Response versus Restart Temperature (2 of 3)

CI	Server(s)	Special Behavior on Warm Restart	Special Behavior on Cold Start or Cold Restart
	EcInPolling	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 EcInReqMgr. Requests remaining in the polling directory are sent as new requests.
	EcInGUI EcInEmailGWServer	N/A	N/A
SBSRV	EcSbSubServer	SBSRV performs all unprocessed actions (including re-submissions of ACQUIRE requests to the SDSRV), and resume accepting new event notifications from the SDSRV.	SBSRV removes 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.
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 deletes all (prior) request information from its databases.

Table 4.11-3. Server Response versus Restart Temperature (3 of 3)

CI	Server(s)	Special Behavior on Warm Restart	Special Behavior on Cold Start or Cold Restart
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.
	EcDsStCacheManagerServer	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.
	EcDsStFtpServer	Existing request queues are retained.	Existing request queues are cleared.
	EcDsStDTFServer	N/A	Allocated device will be de-allocated.

- **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 4.11-4. Note that a cell value of N/A means the server either has no clients or the clients do not re-submit requests.

Table 4.11-4. 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, only the part of the re-submitted request that has not been completed is re-processed.
	EcDpPrDeletion	Requests are submitted synchronously. If the entire request is re-submitted by a client, only the part of the re-submitted request that has not been completed is re-processed.
INGST	EcInGran EcInReqMgr EcInPolling EcInGUI EcInEmailGWServer	N/A
SBSRV	EcSbSubServer	When SDSRV re-submits the same request, if SBSRV received and buffered it successfully, this second request is not processed. Instead, SBSRV just returns a successful status to the client. When SBSRV re-submits the same request to its action provider, SDSRV, it uses the same rpc ID for this request. As long as SDSRV returns a successful status, this request is removed from the SBSRV side and is not re-submited.
SDSRV	EcDsScienceDataServer EcDsHdfEosServer	All requests are serviced as if they are new requests. Note that since RPC IDs are generated automatically and reproducibly, SDSRV typically recreates 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.
DDIST	EcDsDistributionServer	For synchronous requests, if previously submitted and completed, the request status is returned based on the check-pointed request status. For the other synchronous request, the client request thread is synchronized with the worker thread actually servicing the request. For synchronous requests, a successful status is returned if the server determines the request is already in its database and therefore is a previously submitted request. Otherwise, it is a new asynchronous request.

Table 4.11-4. Server Response for Request Re-submission (2 of 2)

CI	Server(s)	Behavior on Request Re-submission
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.
	EcDsStCacheManagerServer EcDsStFtpServer EcDsStPullMonitorServer EcDsStDTFServer	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.

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 4.11-5. Note that the behavior of a server in the event of a client crash does not vary from client to client.

Table 4.11-5. Server Responses to Client Failures

CI	Server(s)	Behavior on Client Crash
PLANG, PRONG	EcDpPrJobMgmt EcPlSubMgr EcDpPrDeletion	Requests in process are serviced to completion.
INGST	EcInGran EcInReqMgr	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 proceeds to finish all triggered subscriptions until the 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	EcDsStRequestManagerServer EcDsStArchiveServer EcDsStCacheManagerServer EcDsStPullMonitorServer EcDsStFtpServer EcDsSt8MMServer EcDsStDTFServer	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 by End of Contract).

- **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 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 4.11-6 shows exceptions to the general behavior for client submission of restart notification:

Table 4.11-6. Client Restart Notification Exceptions

Client Processes	Server Processes	Restart Notification
PDPS		N/A
EcInGran	EcDsScienceDataServer EcDsStRequestManagerServer	Matches start temperature (Also see Note 1 below)
EcInReqMgr	EcDsStRequestManagerServer	Matches start temperature (See Note 1 below)
EcInGUI	EcDsStRequestManagerServer	Always sent cold. The restart notification is sent the first time the GUI starts to do a media ingest. (See Note 1 below)
EcInPolling EcInEmailGWServer	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)
	EcDsStFtpServer EcDsStDTFServer	N/A (not supported by server)
EcDsStArchiveServer	EcDsStArchiveServer EcDsStCacheManagerServer EcDsStStagingDiskServer	Matches start temperature (Also see Note 1 below)
EcDsStCacheManagerServer EcDsStPullMonitorServer	N/A	N/A
EcDsStStagingDiskServer	EcDsStCacheManagerServer EcDsStStagingDiskServer	Matches start temperature (Also see Note 1 below)
EcDsStFtpServer	EcDsStStagingDiskServer	Matches start temperature (Also see Note 1 below)
	EcDsStPullMonitorServer	N/A (not supported by server)
EcDsStFtpServer EcDsStDTFServer	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 is still 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 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 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 4.11-7:

Table 4.11-7. Server Responses to Client Notification (1 of 2)

CI	Server(s)	Behavior on Cold Notification	Behavior on Warm Notification
PLANG, PRONG	EcDpPrJobMgmt EcPlSubMgr EcDpPrDeletion	N/A	N/A
INGST	EcInGran EcInReqMgr EcInPolling EcInGUI EcInEmailGWServer	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

Table 4.11-7. Server Responses to Client Notification (2 of 2)

CI	Server(s)	Behavior on Cold Notification	Behavior on Warm Notification
STMGT	EcDsStArchiveServer	For partially completed Ingest operations, all files stored are removed. (Partial granules are never permitted in the archive.)	General
	EcDsStStagingMonitorServer EcDsStPullMonitorServer EcDsStFtpServer	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 EcDsStDTFServer	N/A	N/A

Some known limitations within the ECS are:

a.) Data Distribution may be failing and suspending requests it should not fail (but rather, suspend) or should not suspend (but rather, retry).

b.) Requests with many sub-requests can experience timing problems because of nested retries or because one of the requests is suspended.

c.) Coding errors can cause unanticipated fault behavior that is different from what is described above (and such occurrences should be reported as NCRs).

d.) System engineers and designers may have made mistakes in classifying errors (e.g., as fatal versus retry).

e.) Not all ECS applications use the error recovery capabilities of the ECS Sybase interface infrastructure.

4.12 Product Distribution System (PDS) Subsystem Overview

The Product Distribution System (PDS) is a centralized, focused resource for the distribution management of digital products. It receives the customer's order information from the appropriate input source, interfaces with other ECS subsystems to retrieve data for the products to be created, passes the production parameters to the PDS and creates relevant shipping materials for shipment. The PDS Internal Server (PDSIS) is the interface between the ECS and the PDS Standalone (PDSSA) subsystem, which is responsible for actual physical media creation. The PDS includes the following components:

The PDSIS has requirements to support the following capabilities:

1) Accepting digital product requests via V0 GTWAY ODL or the Order Manager Subsystem
2) Requesting digital product data from ECS via a product request parameter file sent to the SDSRV using the SCLI
3) Receiving digital product data from ECS Distribution subsystem via Ftp Push
4) Coordinating PDSSA processing to include detection and resolution of data transfer problems, data flow control, and order recovery
5) Generating packaging and shipping artifacts including packing lists and shipping labels
6) Managing PDSIS processing such as accepting Operator Processing Requests, managing PDSIS data, generating Production Reports, monitoring PDSIS operations, and managing PDSIS Jobs
7) Provide temporary storage for digital data
8) Provide Customer notification that a digital product has been produced and verified

The PDSSA has requirements to support the following capabilities:

1) Acquire Digital Products from disk, resolution and detection of transfer problems, and re-pull data
2) Generate and distribute Digital Products to customers via the following media types:
   CD-Rom, DVD-Rom, High Density 8mm tape, DLT 7000c
3) Provide Generic Production Modules that:
   a. Write one Granule to one Directory. Spans granules across media but do not span files.
   b. Write all granules to one directory. Spans granules across media but do not span files.

4) Label Tape products, label CD-Rom and DVD-Rom and print Jewel Case inserts
    a. Basic Order level information (Order #, Req_ID, date…)
    b. Format printed on Inserts is “Various” via a configured field
5) Remove digital source files upon completion of the media product
6) Manage PDSSA processing including accepting Operator Processing Requests
7) Manage PDSSA Data, Order status, and production reports
8) Manage PDSSA Operations including Operator Interfaces and job management capabilities. This includes system control, job management and tracking, error reporting, fault isolation support and recovery
9) Provide storage for digital data being put on media

Physical media is not available thru MTMGW, Subscription, and ASTER on-demand interfaces.When the ECS-V0 Gateway or OMS receives an order, it separates 8mm tape, DLT, CD-ROM, and DVD delivery requests from other delivery requests. All order requests are entered in the ECS order-tracking database. Deliveries of 8mm tape, DLT, and CD-ROM are handled by the PDS and other requests are forwarded to the ECS Science Data Server.

Figure 4.12-0 diagrams the data flows for the interface. The figure is limited to the physical media orders required by PDS, namely 8mm tape, DLT, CD-ROM, and DVD. This specification uses the term "physical or hard media" to mean these delivery types.

If a physical media order is for other than Landsat-7 products (un-shaded areas in the figure), the gateway forwards the order to the PDS. The interface is socket based and uses ODL message formats. For Landsat-7 products, the V0 Gateway/OMS first forwards physical media orders to the DORRAN at EDC (see shaded area of figure); then when the Gateway receives a validated Product Request from DORRAN, it forwards the Landsat order to the PDS. The V0 Gateway-DORRAN interface messages are documented in the ICD Between ECS and the EDC DAAC, ESDIS document number 423-41-58.

For each order it receives from the V0 Gateway or OMS, the PDS orders the requested data using the SCLI, a command line interface with the ECS Science Data Server. The PDS may break up large orders into smaller sets and orders granules for a request individually. Orders from the PDS for data to be FTP pushed to PDSIS to fill physical media orders do not include Request IDs and are not tracked in the ECS Order Tracking Database. ECS delivers the data to the PDS using its standard ftp push data distribution capability. PDS receives the data on a staging disk via an FTP Push by the Storage Management CSCI of the Data Server Subsystem.

The PDS fills the order and delivers it; then e-mails a data distribution notice (order shipment notification) to the user if configured to do so, and for Landsat-7 orders to DORRAN. It then updates the ECS MSS order-tracking database to complete the status for the physical media order.

Operators can use the PDS operator interfaces to check on the status of physical media requests and control the physical media distribution to users. ECS operator interfaces can be used to obtain status of PDS requests for data from ECS.

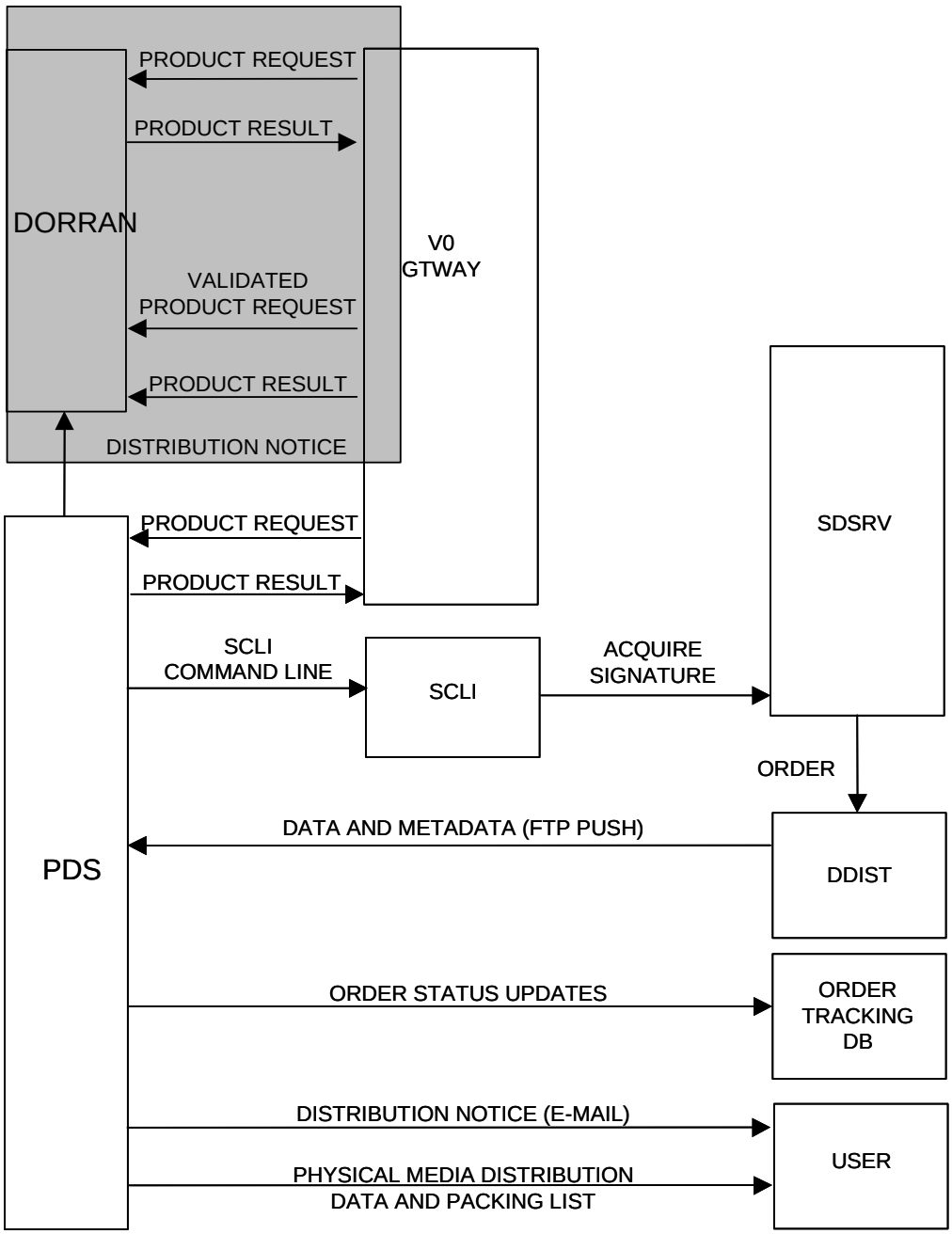


Figure 4.12-0. PDS Data Flow

In Figure 4.12-0 the shaded area applies only to requests for Landsat-7 data. The ECS creation of order tracking records is not shown.

Product Distribution System (PDS) Subsystem Context

Figure 4.12-1 is the PDS Subsystem context diagram. The diagram shows the events sent to the PDS Subsystem and the events the PDS Subsystem sends to other SDPS and CSMS subsystems. Table 4.12-1 provides descriptions of the interface events shown in the PDS Subsystem context diagram.

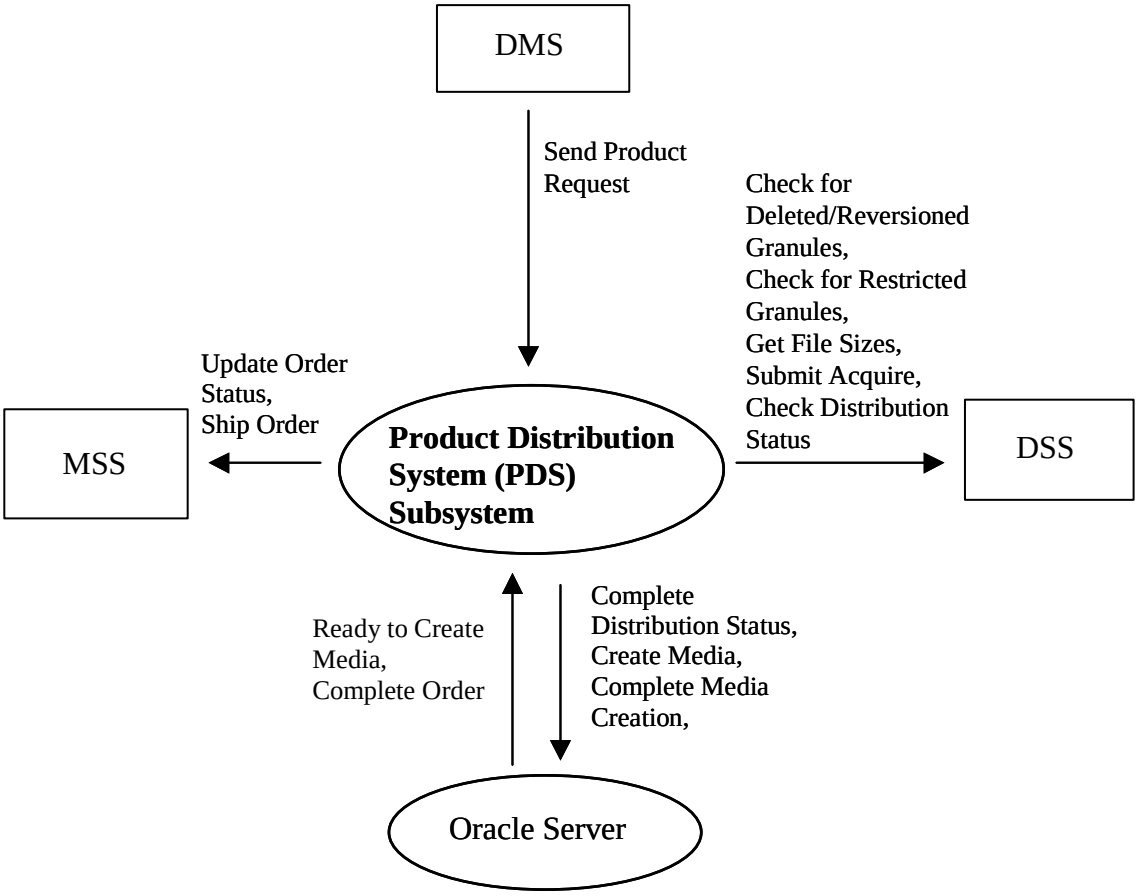


Figure 4.12-1. PDS Subsystem Context Diagram

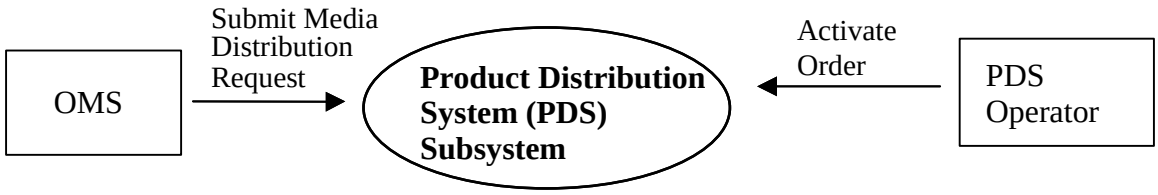


Figure 4.12-1. PDS Subsystem Context Diagram (cont.)

Table 4.12-1. PDS Subsystem Interface Events (1 of 2)

Event	Interface Event Description
Send Product Request	The Product Distribution System (PDS) receives a product request from the **DMS**.
Check for Deleted/Reversioned Granules	The PDS sends a request to the **DSS** to check for reversioned or deleted granules.
Check for Restricted Granules	The PDS sends a request to the **DSS** to check for access to restricted granules.
Get File Sizes	The PDS sends a request to the **DSS** to get file sizes.
Submit Acquire	The PDS submits an acquire request to the **DSS** via the Science Data Server Command Line (SCLI) interface.
Check Distribution Status	The PDS sends a request to the **DSS** to check the status of the distribution of the data.
Complete Distribution Status	The PDS updates the **PDS Oracle database** with the order status.
Create Media	The PDS writes the order information to an ASCII file and starts the require production module, passing in the order information file (the ppf file). The production module writes the order information to the media and updates the order status file.
Complete Media Creation	The PDS updates the **PDS Oracle database** from the order status file with the completed order information and prints tape labels or jewel case inserts.
Ready to Create Media	The PDS retrieves orders ready to process from the **PDS Oracle database** and displays them on the Operator Interface GUI for the PDS Operator to activate. The operator selects an appropriate device with the required media loaded to prepare the order for distribution.
Complete Order	The PDS gets the completed order status from the **PDS Oracle database**. The media type and number of bytes in the order are updated in the MSS order-tracking database.
Update Order Status	The PDS sends a request to the **MSS** to update the order status to “Transferring”.

Table 4.12-1. PDS Subsystem Interface Events (2 of 2)

Event	Interface Event Description
Ship Order	The PDS Operator marks the order on the PDS Operator Interface GUI (PDSISMTMNU) as "Shipped." The PDS prints the shipping label and packing information, sends the order completed Distribution Notification (DN) e-mail message to the user if configured to do so and marks the order as completed in the PDS Oracle database. The PDS updates the order status in the **MSS order-tracking database** to "Shipped".
Activate Order	PDSISMTMNU – the operator activates the order from the PDSIS Operator Interface when the PDSIS is in manual mode.
Submit Media Distribution Request	The **OMS** submits all product distribution requests that specify physical media types to the PDS using the V0 ODL protocol.

PDS Subsystem Structure

The PDS Subsystem is two CSCIs, the Product Distribution System Internal Server (PDSIS) and the Product Distribution System Stand-Alone (PDSSA) and a COTS process (the Oracle Enterprise Server used with Oracle Forms and Reports).

- The Product Distribution System Internal Server (PDSIS) is a software configuration item. The PDS manages the distribution of digital products from the ECS. It receives customer orders via the Version 0 Gateway (V0 GTWAY) or the Order Manager Subsystem (OMS). The PDSIS interfaces with the ECS to retrieve data for the products to be created and creates relevant shipping materials.
- The Product Distribution System Standalone (PDSSA) is a software configuration item. The PDSSA interfaces with the PDSIS to retrieve data for the products to be created and creates relevant shipping materials for shipment.

Use of COTS in the PDS Subsystem

Note: The following RogueWave Libraries are currently delivered with custom code as static libraries. A separate installation of dynamic libraries is no longer required.

- The Rogue Wave's Tools.h++ class libraries are used by the PDS only in the FTP password encryption program.

Other COTS products are also utilized to support the PDS Subsystem.

- Oracle Enterprise Server

  Oracle provides access for the PDS to insert, update, and delete PDS database information. The Oracle Enterprise Server must be running during operations for the PDS to execute, search, and update requests on the PDS databases.

- Oracle Forms 4.5 and Oracle Reports 2.5

  Oracle Forms are used for the PDS GUIs and Oracle Reports are used to create reports on PDS operation and in producing CD/DVD jewel case inserts.

- Sybase Adaptive Server Enterprise (ASE)

  Sybase's ASE provides access for the PDS to select and update database information. The Sybase ASE must be running during operations for the PDS to execute, search, and update requests in the MSS order-tracking database. The Sybase ASE must be running during operations for the PDS to execute and search on the STMGT and SDSRV databases.

- GNU Make 3.75

  The GNU Make is used for building mkisofs.

- Land Analysis System 7.3

  The Land Analysis System (LAS) is a system designed to provide the user with the tools necessary to perform image processing and analysis efficiently. Providing these tools, LAS encompasses each step--from ingestion of the data to the generation of output products.

  LAS has been developed through a cooperative effort between the GSFC and the EDC.

- mkisofs 1.13

  The mkisofs is used for pre-mastering iso9660 file systems, which are used on CDs.

- HDF 4.1r3

  At its lowest level, HDF is a physical file format for storing scientific data. At its highest level, HDF is a collection of utilities and applications for manipulating, viewing, and analyzing data in HDF files. Between these levels, HDF is a software library that provides high-level APIs and a low-level data interface.

- a2ps

  The a2ps creates postscript data from ASCII files.

PDSIS Error and Debug Log Handling

PDSIS maintains daily logs for both Error Logs and Debug logs for troubleshooting purposes. The error log may give slightly more information on the nature of the problem then the PDSIS Error screen & gives the operator a history of daily problems.

Error Logs:

/usr/local/pdsis[_mode]/logs/errors

The error logs have the following format for each message:

```
CLASS NAME: EcsDataRequest
METHOD    : ScliRequest
ORDER ID  : YEA0307140001
UNIT NUM  : 2
```

```
TRIES     : 1
ERROR TEXT: Error executing SCLI
ERROR NUM : 4021
COMMENT   : There was an error executing the SCLI for order
number = YEA0307140001, unit = 2
DATE/TIME : Mon Jul 14 13:47:12 EDT 2003
```

There are error logs for error messages generated by the PDSIS cron jobs. These are located in the /usr/local/pdsis[_mode]/logs directory.

Debug Logs:

The debug log logs all pertinent server information as the cron runs. Nominal/successful processing is logged in this file so they can grow large. Problems may be logged below the level of a declared error in this debug file, which can provide the operator with the health of the PDSIS server:

/usr/local/pdsis[_mode]/logs/debug

The debug logs have the following format for each message:

```
CLASS NAME: ProcessMail
METHOD    : CheckStageManagment
TEXT      : Received status = null for order number =
YEA0306090002, unit = 1
DATE/TIME : Mon Jul 14 00:00:11 EDT 2003
```

Processing Logs:

Other logs are included under /usr/local/pdsis[_mode]. They are:

- The in_msg logs subdirectory contains incoming to PDSIS ODL messages
- The out_msg logs subdirectory contains outgoing ODL product result messages from PDSIS
- The sockets subdirectory contains information incoming on the socket PDSIS is listening for requests on
- The orders subdirectory contains the order ODL file, and the parameter and request files for each granule in the order used as input to the SCLI when submitting acquire requests for the data

4.12.1 Product Distribution System Internal Server (PDSIS) Software Description

4.12.1.1 PDSIS Functional Overview

The PDSIS manages the distribution of digital products from the ECS. It receives customer orders via the V0 GTWAY. When the V0 GTWAY receives an order, it separates 8mm tape, DLT, CD-ROM and DVD delivery requests from other delivery requests. All order requests are entered in the ECS MSS order-tracking database. Deliveries of 8mm tape, DLT, and CD-ROM are handled by the PDSIS.

4.12.1.2 PDSIS Context

Figure 4.12-2 is the PDSIS CSCI context diagram. The diagram shows the events sent to other CSCIs or CSCs and the events the PDSIS CSCI receives from other CSCIs and CSCs. Table 4.12-2 provides descriptions of the interface events shown in the PDSIS CSCI context diagram.

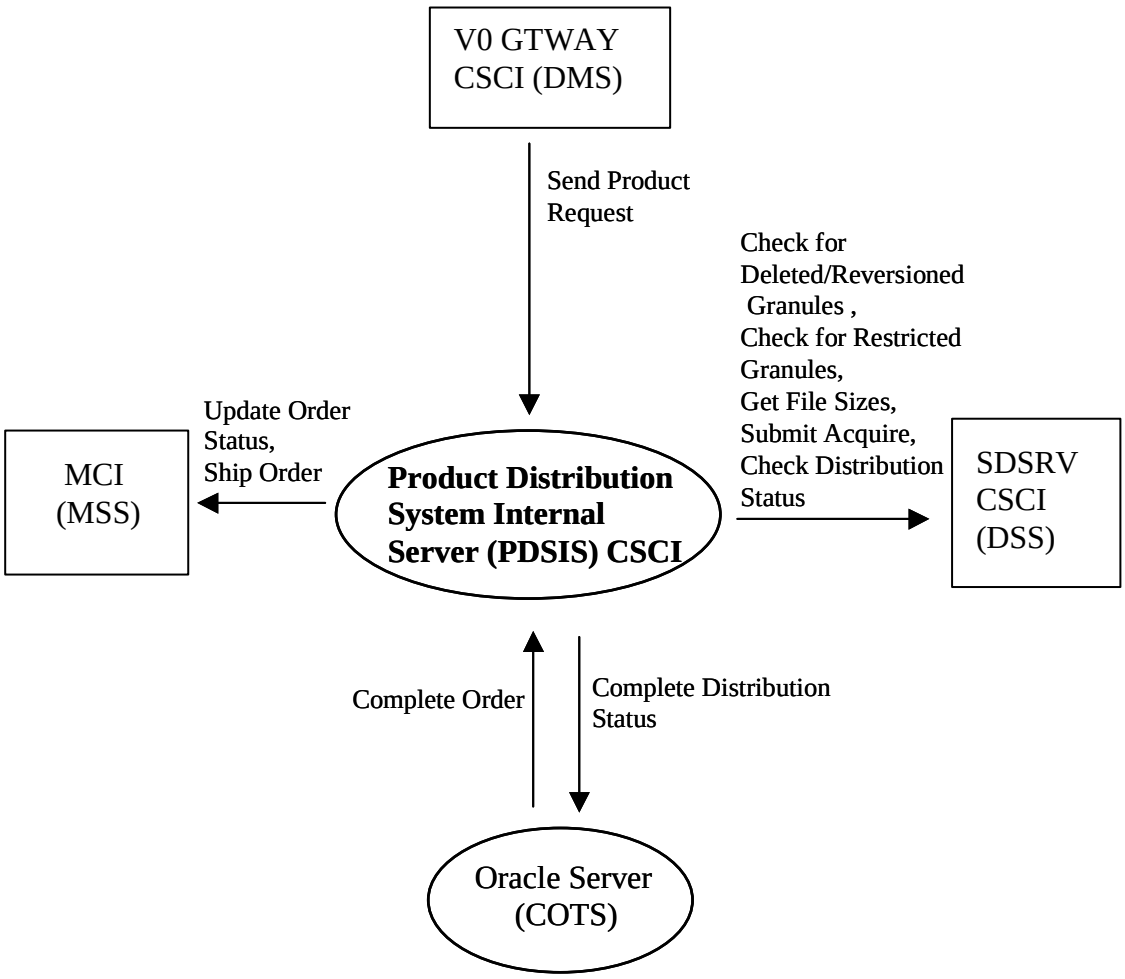


Figure 4.12-2. PDSIS CSCI Context Diagram

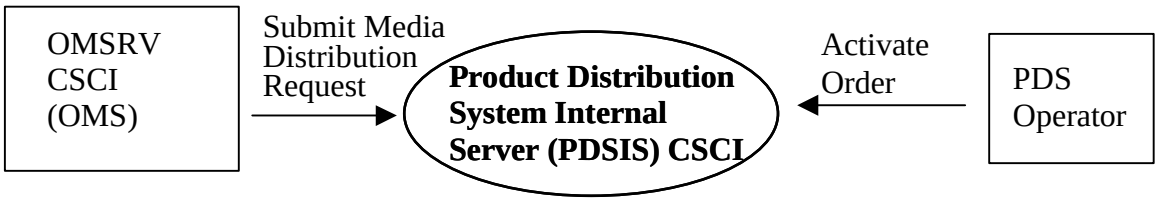


Figure 4.12-2. PDSIS CSCI Context Diagram (cont.)

Table 4.12-2. PDSIS CSCI Interface Events

Event	Interface Event Description
Send Product Request	The Product Distribution System Internal Server (PDSIS) CSCI receives a product request from the **V0 GTWAY CSCI**.
Check for Deleted/Reversioned Granules	The PDSIS CSCI sends a request to the **SDSRV CSCI** to check for reversioned or deleted granules.
Check for Restricted Granules	The PDSIS CSCI sends a request to the **SDSRV CSCI** to check for access to restricted granules.
Get File Sizes	The PDSIS CSCI sends a request to the **SDSRV** CSCI to get file sizes.
Submit Acquire	The PDSIS CSCI submits an acquire request to the **SDSRV** CSCI.
Check Distribution Status	The PDSIS CSCI sends a request to the **SDSRV** CSCI to check the status of the distribution of the data.
Complete Distribution Status	The PDSIS CSCI updates the **PDS Oracle database** with the order status.
Complete Order	The PDSIS CSCI gets the completed order status from the **PDS Oracle database**. The media type and number of bytes in the order are updated in the MSS order-tracking database.
Update Order Status	The PDSIS CSCI sends a request to the **MCI** CSCI to update the order status.
Ship Order	The PDSIS CSCI Operator marks the order on the PDS Operator Interface GUI as “Shipped.” The PDSIS CSCI prints the shipping label and packing information, sends the order completed Distribution Notification (DN) e-mail message to the user if configured to do so and marks the order as completed in the PDS Oracle database. The PDSIS CSCI updates the order status in the **MSS order-tracking database** to “Shipped”.
Activate Order	PDSISMTMNU – the operator activates the order from the PDSIS Operator Interface when the PDSIS is in manual mode.
Submit Media Distribution Request	The **OMSRV CSCI** submits requests for product distribution requests on physical media to the PDSIS CSCI.

4.12.1.3 PDSIS Architecture

Figure 4.12-3 is the PDSIS CSCI architecture diagram. The diagram shows the events sent to the PDSIS CSCI processes and the events the PDSIS CSCI processes send to other processes.

The PDSIS CSCI is one SDPS process, the Product Distribution System Internal Server (PDSIS) and a COTS process (the Oracle Server).

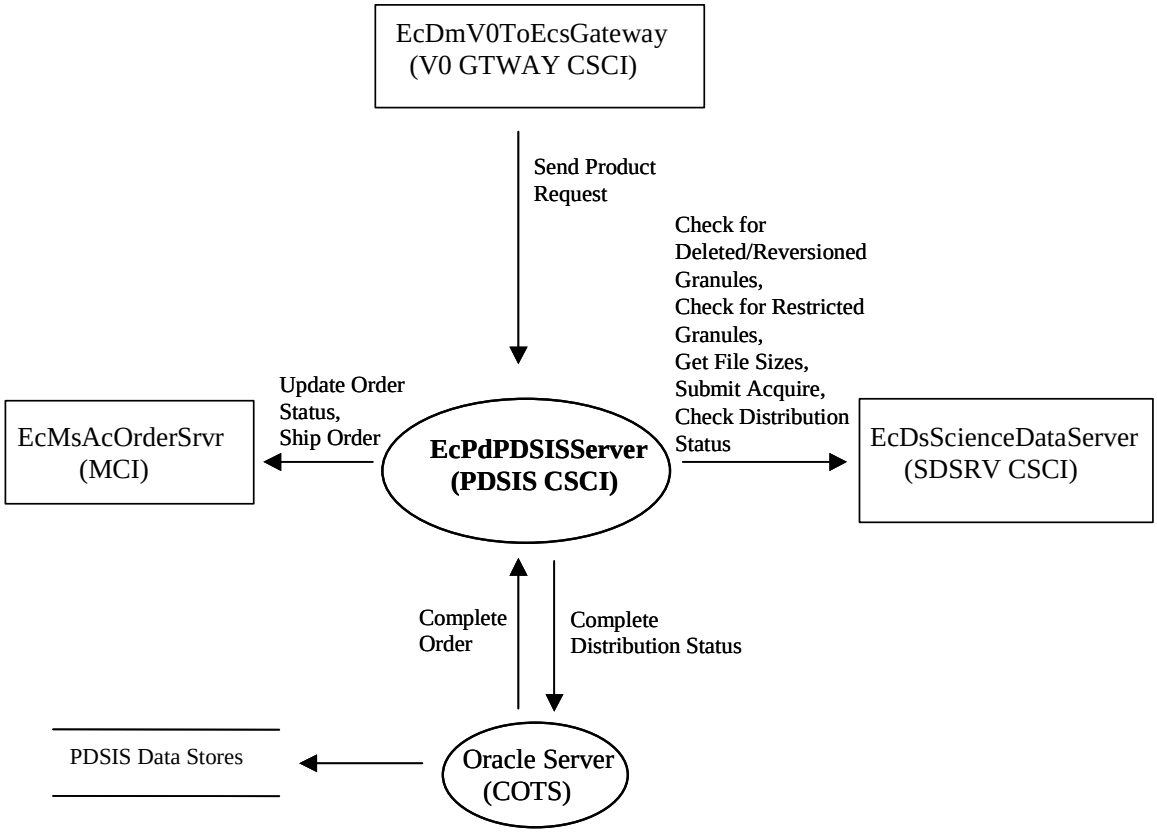


Figure 4.12-3. PDSIS CSCI Architecture Diagram

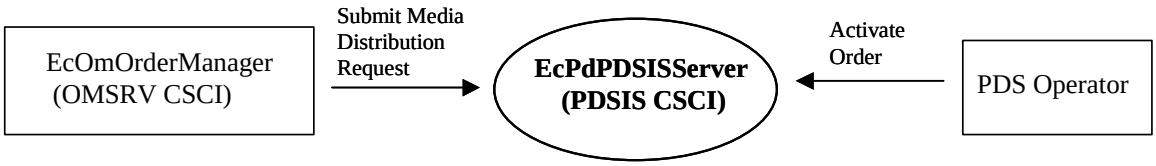


Figure 4.12-3. PDSIS CSCI Architecture Diagram (cont.)

4.12.1.4 PDSIS Process Descriptions

Table 4.12-3 provides descriptions of the processes shown in the PDSIS CSCI architecture diagram.

Table 4.12-3. PDSIS CSCI Processes (1 of 2)

Process	Type	COTS/ Developed	Functionality
EcPdPDSISServer	Server	Developed	Operators can use the PDS operator interfaces (PDSIS Operator Interface, PDSISMTMNU, and PDSISMTOIX) to check on the status of physical media requests and control the physical media distribution to users. ECS operator interfaces can be used to obtain the status of PDS requests for data from ECS.
Oracle	Server	COTS	The Oracle Enterprise Server provides access for the PDS to insert, update, and delete PDSIS database information.
PDSIS SCLI Cron	Cron	Developed	The PDSIS SCLI cron submits acquire requests to the DSS for each requested granule.
PDSIS status cron	Cron	Developed	The PDSIS status cron checks the status of an order at each step. It checks that data has been FTP pushed from the DSS, and checks that requests have been completed by the PDSSA component. At each step, the necessary updates are made to the order in the MSS database.
PDSIS cleanup Cron	Cron	Developed	The PDSIS cleanup job completes jobs that are marked ready to ship or rejected by printing paperwork for the order and sending notifications to the requestor.
PDSIS Operator Interface	Oracle Form	Developed	The PDSIS OI is used to display incoming product requests, reject orders, activate orders when PDSIS is set to manual mode, and ship completed orders.
Reset Order Program	Java Program with start script	Developed	The EcPdPDSISResetOrder script runs the ResetOrder program, which allows a user to restart the SCLI acquire process for an order. The script resets orders so they are restaged to the PDSIS staging area.
PDSISMTMNU	Oracle Form	Developed	The PDSIS Operator Interface GUI. This interface is used to update and view PDSIS database tables.

Table 4.12-3. PDSIS CSCI Processes (2 of 2)

Process	Type	COTS/ Developed	Functionality
PDSISMTOIX	Oracle Form	Developed	An Oracle Form application that displays jobs in PDSIS. It is used to activate the staging of granules when PDSIS is run in the manual mode. It is used to mark jobs as shipped when orders complete in PDSSA and updated to completed by cron jobs in the PDSIS.
PdsReports	Applicati on	Developed	A java program used to create reports on jobs in the system.
EcPdPDSISCleanup	Script	Developed	A script used to create a cron job that cleans up and/or archives old log files.

4.12.1.5 PDSIS Process Interface Descriptions

Table 4.12-4 provides descriptions of the interface events shown in the PDSIS CSCI architecture diagram.

Table 4.12-4. PDSIS CSCI Process Interface Events (1 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
Request Management Services	At system startup and for restarts	*Process:* EcPdPDSISServer	*Script:* EcPdPDSISServerStart	**System startup and shutdown -** Please refer to the release-related, current version of the Mission Operations Procedures for the ECS Project document (611) and the current ECS Project Training Material document (625), identified in Section 2.2.1 of this document.
Send Product Request	Per user request	*Process:* EcPdPDSISServer *Library:* Pdsis.jar *Class:* ECSPDSServer	*Process:* EcDmV0ToEcsGateway *Library:* Pdsis.jar *Class:* ECSPDSServer	The **EcDmV0ToEcsGateway** sends a product distribution request to the EcPdPDSISServer.
Check for Deleted/Reversio ned Granules	Per granule	SDSRV database (DsMdGranule table) via JDBC call	***Process:*** EcPdPDSISServer *Libraries:* Pdsis.jar, Ecs package *Class:* EcsDataRequest	The EcPdPDSISServer (PDSIS SCLI cron) receives a request from the **V0 GTWAY CSCI** to check for reversioned or deleted granules.

Table 4.12-4. PDSIS CSCI Process Interface Events (2 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
Check for Restricted Granules	Per granule	*Process:* EcDsScienceDataServer (by executing stored procedure: ProcGetAccessPrivilege)	*Process:* EcPdPDSISServer *Libraries:* Pdsis.jar, Ecs package *Class:* EcsDataRequest	The EcPdPDSISServer (PDSIS SCLI cron) sends a request to the **EcDsScienceDataServer** to check for access to restricted granules.
Get File Sizes	Per granule	SDSRV database (DsMdGranule table) via JDBC call	*Process:* EcPdPDSISServer *Libraries:* Pdsis.jar, Ecs package *Class:* EcsGranuleSize	The EcPdPDSISServer sends a request to the **EcDsScienceDataServer** to get file sizes.
Submit Acquire	Per granule	*Process:* EcDsScienceDataServer *Library:* DsCl *Classes*: DsClESDTReferenceCollector, DsClRequest	*Process:* EcPdPDSISServer *Libraries:* Pdsis.jar, Ecs package *Class:* Scli Interface	The EcPdPDSISServer submits an acquire request to the **EcDsScienceDataServer**.
Check Distribution Status	Per granule	STMGT database (DsDdRequest, DsDdParameterList tables) via JDBC call	*Process:* EcPdPDSISServer *Libraries:* Pdsis.jar, Ecs package *Class:* EcsSmsStatus	The EcPdPDSISServer sends a request to the **EcDsScienceDataServer** to check the status of the distribution of the data.
Complete Distribution Status	Per user order	*Process:* Oracle Server (COTS)	*Process:* EcPdPDSISServer *Libraries:* Pdsis.jar, Ecs package *Class:* EcsSmsStatus	The EcPdPDSISServer updates the **PDS Oracle database** with the order status.

Table 4.12-4. PDSIS CSCI Process Interface Events (3 of 3)

Event	Event Frequency	Interface	Initiated By	Event Description
Complete Order	Per user order	*Process:* EcPdPDSISServer *Libraries:* Pdsis.jar, Pds package *Class:* CompleteOrder	*Process:* Oracle Server (COTS)	The EcPdPDSISServer gets the completed order status from the **PDS Oracle database**. The media type and number of bytes in the order are updated in the MSS order-tracking database soon after.
Update Distribution Status	One update per order	*Data Base Table:* Pdsis_units_tbl	*Process:* Oracle Server (COTS)	The Oracle Server marks the order as completed in the PDS Oracle database.
Update Order Status	One per order	*Process:* EcMsAcOrderSrvr (By executing stored procedure: ProcUpdPdsRequest)	*Process:* EcPdPDSISServer *Libraries:* Pdsis.jar, Ecs package *Class:* EcsMssUpdate	The EcPdPDSISServer (PDSIS Status cron) sends a request to the **EcMsAcOrderSrvr** to update the order status.
Ship Order	One per order	*Process:* EcMsAcOrderSrvr (By executing stored procedure: ProcUpdPdsRequest)	*Process:* EcPdPDSISServer *Libraries:* Pdsis.jar, Pds package *Class:* Ship Order	The EcPdPDSISServer (PDSIS Cleanup cron) Operator marks the order on the PDS Operator Interface GUI as “Shipped.” The EcPdPDSISServer prints the shipping label and packing information, sends the order completed Distribution Notification (DN) e-mail message to the user if configured to do so and marks the order as completed in the PDS Oracle database. The EcPdPDSISServer updates the order status in the **MSS order-tracking database** to “Shipped”.
Submit Media Distribution Request	One or more per service request	*Process:* EcPdPDSISServer *Library:* Pdsis.jar *Class:* ECSPDSServer	*Process:* EcOmOrderManager *Library:* V0XV0Staticlibs *Class:* OmPDSIf	The **EcOmOrderManager** submits requests for product distribution requests on physical media to the EcPdPDSISServer.

4.12.1.6 PDSIS CSCI Data Stores

Table 4.12-5 provides descriptions of the data stores shown in the PDSIS CSCI architecture diagram.

The data to be distributed that is FTP pushed from the DDIST/STMGT subsystems is stored at the /pdsis/pdsis[_mode]/staging/orders/<order number>/<subdirectory> location, where mode is blank for OPS, _ts1 for TS1, _ts2 for TS2 mode. <order number> is the PDS order number, and <subdirectory> is a directory named with a sequential number for each granule in the order, starting at 1.

Table 4.12-5. PDSIS CSCI Data Stores

Data Store	Type	Functionality
pdsis_orders	Oracle	This database table is used for storing order information. This table is owned by PDSSA and resides in the PDSIS table space.
Pdsis_units	Oracle	This database table is used for storing the product unit information. This table is owned by PDSSA and resides in the PDSIS table space.
Pdsis_address	Oracle	This database table is used for storing the contact, billing and shipping address information. This table is owned by PDSSA and resides in the PDSIS table space.
Odl_pds_lookup	Oracle	This database table is used for storing ODL message parameter to PDS mapping information. This information can be accessed every time an ODL PRODUCT_RERQUEST comes in. It is used to lookup the DATASET_ID, MEDIA_TYPE, MEDIA_FORMAT values from the PRODUCT_REQUEST ODL message to get the appropriate prod_code and output_specs. This table is owned by PDSSA and resides in the PDSIS table space.
Lookup_prodinfo	Oracle	This database table is used for storing information that needs to be passed to the pdt_pdsinfo table based on the product code (product specific information). PDSIS looks up the required information by using the prod_code determined from the odl_pds_lookup_tbl table. PDSIS can be accessed when the DAN is received from ECS. This table is owned by PDSSA and resides in the PDSIS table space.
Lookup_outspecinfo	Oracle	This database table is used for storing output specification information passed to PDS. All data except media_size is passed to PDS. This table is owned by PDSSA and resides in the PDSIS table space.
Pdsis_serverconfig	Oracle	This database table is used for storing PDSIS server configuration information. This table is owned by PDSSA and resides in the PDSIS table space.
Pdsis_userconfig	Oracle	This database table is used for storing PDSIS server remote account information. There **shall not** be any GUI interface to this table because of the RISK of compromising a user name and password. This table is owned by PDSSA and resides in the PDSIS table space.

4.12.2 Product Distribution System Standalone (PDSSA) Software Description

4.12.2.1 PDSSA Functional Overview

The PDS Stand Alone is a centralized, focused resource for the distribution management of digital products. It receives the customer's order information from the PDS Information Server (PDSIS), interfaces with PDSIS to retrieve data for the products to be created, produces the required products, and allows access to the PDSIS to monitor the production status.

4.12.2.2 PDSSA Context

The PDSSA does not have a context diagram, as it is a standalone CSCI.

4.12.2.3 PDSSA Architecture

Figure 4.12-4 is the PDSSA CSCI architecture diagram. The diagram shows the events sent to other processes (internal only) and returned to the PDSSA CSCI.

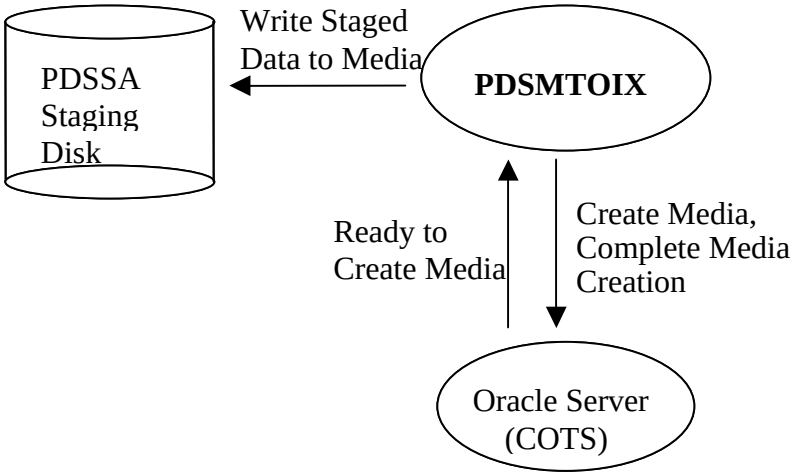


Figure 4.12-4. PDSSA CSCI Architecture Diagram

4.12.2.4 PDSSA Process Descriptions

Table 4.12-6 provides the description of the process shown in the PDSSA architecture diagram.

Table 4.12-6. PDSSA CSCI Processes (1 of 2)

Process	Type	Source	Functionality
PDS Operator Interface (PDSOI)	Oracle Form	Developed	The PDS Operator Interface (PDSOI) module resides on machines capable of generating customer products and contains the Oracle Forms and Reports runtime engine. The operators start up the OI using a shell script (pdsoi_prod.sh) or using an alias that is set up, start_pds. This script then makes a connection to the Oracle database. This connection is made using the Oracle SQL*Net communication mechanism. The script insures that all operators log onto Oracle with the same usercode/password. This usercode has only privileges against those tables used specifically by PDS. Any data needed to produce the product is included in these tables. The PDSOI module generates several reports dealing with order status and jewel case inserts. Some of the information used is stored in the Oracle database tables and the Oracle Report generator is used to create these reports. The Oracle report modules reside on the PDS machine and use the SQL*Net connection to get any data needed for these reports. The user has, as an option, the capability to receive products via FTP. The PDSOI module sends an e-mail message to the user when the product has been generated and delivered to an FTP retrievable directory. This e-mail is sent using the e-mail address provided in the PDT_PDSINFO table using the UNIX sendmail utility on the PDS machine. The support maintenance for the OI module may require an Oracle usercode/password that is different from the general one used to run the OI module. This is to add security so any operator is not able to modify these maintenance tables or perform functions that require additional training and knowledge. The operator starts the maintenance module using a shell script (pdsmaint_prod.sh) or using an alias that is setup, pds_maint. These modules reside on the PDS machine as well. The communication philosophy employed allows for multiple copies of the PDS code to reside anywhere as long as they have SQL*Net and the physical communication means to connect to the PDS machine.

Table 4.12-6. PDSSA CSCI Processes (2 of 2)

Process	Type	Source	Functionality
Pds_insert.cron	Cron	Developed	The PDSSA insert cron adds jobs to the PDSOI.
Pds_jobs.cron	Cron	Developed	The PDSSA job cron directs the production of media. When a job is activated via the PDSOI, the job cron monitors the production, checks for bad statuses, and updates job status.
Pds_completion.cron	Cron	Developed	The PDSSA completion cron runs after the operator marks the job as completed in the PDSOI. The status is updated so the PDSISOI can be notified that the media has been created.
Jobmonitor	Application	Developed	A Perl script used to monitor jobs sent to the Rimage.
PDSMNMSM	GUI	Developed	The PDS Maintenance GUI (oracle form) is used to update and view PDSSA database tables. The GUI has a reset order option to allow the operator to restage the data and remake the media.
ReprintGUI	GUI	Developed	A script used to reprint shipping labels and packing lists.
Pdscleanup	Application	Developed	A script that creates a cron job that cleans up and/or archives old log files.
EcPdSaQCGui	Application	Developed	Tcl/Tk and Perl scripts used to quality check finished media.
EcPdSaPasswdChg	Application	Developed	A process used to change encrypted pds passwords used for FTP from DSS to PDSIS and from PDS to the Rimage.
Oracle	Server	COTS	The Oracle Enterprise Server provides access for the PDS to insert, update, and delete PDS database information.

4.12.2.5 PDSSA Process Interface Descriptions

Table 4.12-7 provides descriptions of the interface events shown in the PDSSA CSCI architecture diagram.

Table 4.12-7. PDSSA Process Interface Events (1 of 2)

Event	Event Frequency	Interface	Initiated By	Description
Create Media	One per product request	*Process:* Oracle Server (COTS)	*Process:* PDSMTOIX *Library:* PDSOI	The PDSMTOIX writes the order information to an ASCII file and starts the require production module, passing in the order information file (the ppf file). The production module writes the order information to the media and updates the order status file.
Complete Media Creation	One per product request	*Process:* PDSMTOIX pds_jobs.cron	*Process:* Oracle Server (COTS)	The PDSMTOIX updates the **PDS Oracle database** from the order status file with the completed order information and prints tape labels or jewel case inserts.
Ready to Create Media	One per product request	*Process:* PDSMTOIX pds_insert.cron	*Process:* Oracle Server (COTS)	The PDSMTOIX retrieves orders ready to process from the **PDS Oracle database** and displays them on the Operator Interface GUI for the PDS Operator to activate. The operator selects an appropriate device with the required media loaded to prepare the order for distribution.

Table 4.12-7. PDSSA Process Interface Events (2 of 2)

Event	Event Frequency	Interface	Initiated By	Description
Ready to Create Media	One per product request	*Process:* PDSMTOIX pds_insert.cron	*Process:* Oracle Server (COTS)	The PDSMTOIX retrieves orders ready to process from the **PDS Oracle database** and displays them on the Operator Interface GUI for the PDS Operator to activate. The operator selects an appropriate device with the required media loaded to prepare the order for distribution.
Write Staged Data to Media	One per order request	PDSSA Staging Disk	*Process:* PDSMTOIX *Class:* Production Module	The PDSMTOIX writes the granule(s) requested by the user to the appropriate media from the PDSIS staging disk area.

4.12.2.6 PDSSA Data Stores

The data to be distributed that is copied from the PDSIS staging area is stored at the /pdssa/assembly[_mode] /<PDS job Id>/<order number> location, where mode is blank for OPS, _ts1 for TS1, _ts2 for TS2 mode. <order number> is the PDS order number.

Table 4.12-8. PDSSA CSCI Data Stores (1 of 2)

Data Store	Type	Functionality
PDT_PDSINFO	Oracle	The interface between the PDSIS and the PDS consists of the database table PDT_PDSINFO. The PDT_PDSINFO table contains all PDS orders that are either currently or historically in the system.
PWT_PDS_WORK	Oracle	The PWT_PDS_WORK_TBL is very similar to the PDT_PDSINFO table, except that it only holds records that are currently in the queue for the PDS subsystem. Records are insert into this table when they have a PDT_PDSINFO.STATUS of “Q.” All the processing by the PDSOI is done against the PWT_PDS_WORK_TBL table. When a unit is completed then updates are made to the PDT_PDSINFO table for the corresponding unit and the PWT_PDS_WORK_TBL record can be deleted.

Table 4.12-8. PDSSA CSCI Data Stores (2 of 2)

Data Store	Type	Functionality
PJT_PDSINFO_JOBS	Oracle	The PJT_PDSINFO_JOBS table contains the jobs currently in the system. This schema is used for the display of the PDSOI Main Window.
PVT_PRCDTBL	Oracle	The PVT_PRCDTBL_TBL table holds the product codes that use the PDS to produce products. Any product code used in the PDT_PDSINFO table must match one in this table.
OUT_OTSPTBLV	Oracle	The OUT_OTSPTBLV_TBL table holds the specifications of the output that can be produced by the PDS. Any output specifications used in the PDT_PDSINFO table must match one in this table.
PJC_PDS_JEWEL_CASES	Oracle	The PJC_PDS_JEWEL_CASES table is used by the software in passing information to the Oracle report used for creating the jewel case inserts.
PTT_PDS_PPF	Oracle	The PTT_PDS_PPF_TBL is the PPF Building table. This table contains the definitions for the entries included in building the PPF file for the PDS subsystem.
PXT_PRINTERS	Oracle	The PXT_PRINTERS_TBL is the printer table. This table contains the valid list of printers to be used by the PDSOI. It is used to validate printer selections in the PDSOI.
JLT_JOBLIMITS	Oracle	The JLT_JOBLIMITS_TBL table is the job limits table. This table is used by the PDSOI to determine the number of units to be run at a time. The number of units can be machine dependent. The ppf file is generated using numbers from this table.
MCT_MACHINFO	Oracle	The MCT_MACHINFO_TBL table is the PDS machine table. This table identifies the logical PDS machines and machine dependent information used by the PDSOI.
PGT_PDS_PPF_TMP	Oracle	This table is a temporary table used when generating the .ppf file for the product generation software. This table only contains values when a .ppf file is being created.
LKT_LOOKUPS	Oracle	The LKT_LOOKUPS_TBL is a list of lookups used for processing. This table is used by the PDSOI for things like defaults, pick list, etc.
APT_APP_ERRORS	Oracle	This table is used to capture errors encountered during processing.

4.12.2.7 PDS Hardware Section

See DIPHW for description of PDS HW. PDS is part of the Distribution and Ingest Peripheral (DIP) in the DDIST section (4.1.2) of this document.

4.13 Spatial Subscription Server Subsystem Overview

The Spatial Subscription Server (SSS) subsystem is the principal means by which users can establish standing orders for data. Users enter subscriptions for specific data types using a GUI or command line interface (CLI). A subscription may be qualified by specifying one or more constraints on the metadata of matching granules. This includes the capability of qualifying the subscription spatially by specifying a geographic area over which the data was collected. A subscription has one or more associated actions such as data distribution, email notification, Data Pool insertion, or bundling, i.e. adding a granule to an Order Manager bundle.

In addition to the subscription creation components, the SSS subsystem is comprised of a Sybase database and four runtime servers: an event server to match subscriptions with granule events, an action server to execute the actions of matched subscriptions, a recovery server to restart stalled events or actions, and a deletion server to clean up the database.

Spatial Subscription Server Subsystem Context

Figure 4.13-1 is the Spatial Subscription Server Subsystem context diagram. The diagram shows the events sent to the SSS Subsystem and the events the SSS Subsystem sends to other subsystems. Table 4.13-1 provides descriptions of the interface events shown in the SSS Subsystem context diagram.

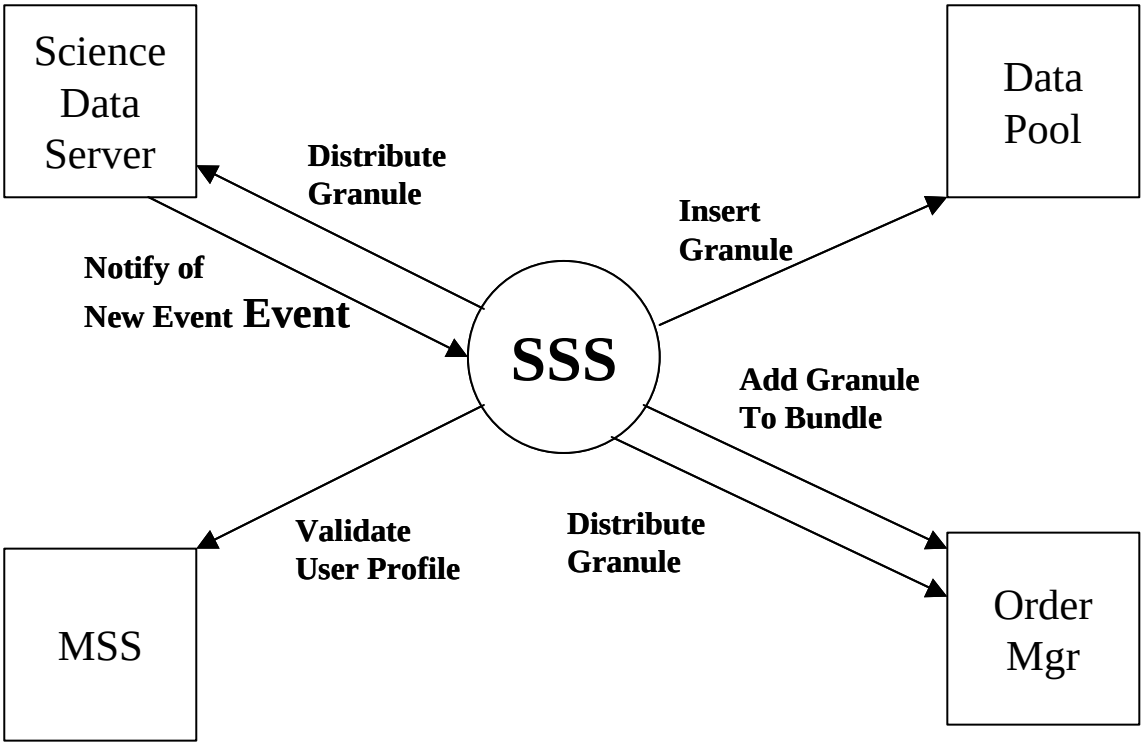


Figure 4.13-1. Spatial Subscription Server Subsystem Context Diagram

Table 4.13-1. Spatial Subscription Server Subsystem Interface Events

Event	Interface Event Description
Notify of New Event	The Science Data Server notifies SSS when a new granule has arrived that could match a subscription. It does this via a trigger in the SDS database that inserts into the SSS event queue, a table in the SSS database.
Distribute Granule	SSS will notify Science Data Server, via its command line interface (SCLI), when a matched subscription has an acquire action, if the SSS action driver was configured for acquire mode of SCLI.
Validate User Profile	When a subscription is created or modified, SSS will verify that the subscription owner or the recipient of an acquire action has a valid user profile in the MSS database.
Insert Granule	SSS will insert a granule into the Data Pool when a matched subscription has an associated Data Pool action. It does this by inserting an entry into DlInsertActionQueue, a table in the Data Pool database.
Add Granule to Bundle	SSS will cause a granule to be added to an OMS bundle when a matched subscription has an associated OMS bundling order. This is accomplished via a call to a stored procedure in the OMS database.
Distribute Granule	SSS will notify Order Manager, via a call to a stored procedure in the OMS database, when a matched subscription has an acquire action, if the SSS action driver was configured for acquire mode of OMS.

4.14 Data Pool Subsystem Overview

NOTE: The Data Pool subsystem was developed and delivered as Operational Support Software under the ECS Statement of Work, and therefore does not have the same documentation format and requirements as other ECS subsystems. An overview of the Data Pool subsystem is provided here for reference purposes.

The Data Pool is a large online cache of data at each DAAC. Data in the Data Pool may be either data from the ECS archive, or non-ECS data selected by the DAAC. Science, metadata (in xml format), and browse files (in jpg format) are stored in the Data Pool. The Data Pool subsystem consists of the following components and supporting utilities:

1. Data Pool Insert: inserts ECS or non-ECS data into the Data Pool. ECS data is copied from AMASS into the Data Pool, based on either Data Pool insert actions assigned from the Spatial Subscription Server, or on ECS granule ids submitted by the Batch Insert Utility. The Data Pool database inventory is updated for each granule inserted in the Data Pool. Data Pool Insert consists of four subcomponents: a) the Data Pool Action Driver (EcDlActionDriver): a C++ executable which schedules Data Pool insert actions based on a queue of Data Pool insert actions populated by the Spatial Subscription Server or the Batch Insert Utility; b) the Data Pool Insert Utility (EcDlInsertUtility), a java executable which manages the insert of data into the Data Pool, and which updates the Data Pool database inventory; c) the Data Pool copy server (EcDlQuickServer), a C++ executable which is installed on each AMASS host and which checks AMASS file status and performs the actual copy operations from AMASS to the Data Pool for ECS data; and d) the Data Pool XML generator (EcDlM2XTApp), a java executable which translates ECS granule metadata from the Science Data Server database into XML, for storage in the Data Pool directories.
2. Data Pool Cleanup (EcDlCleanupDataPool): a perl utility which cleans expired granules from the Data Pool directories and database. This utility normally runs as a cron job. The utility may also be used to report on and correct inconsistencies between the Data Pool directories and the database.
3. Data Pool Web Access (EcDlWebAccess): a java-based web application, which runs with the apache web server and related COTS. The Data Pool Web Access application allows end-users to perform drill-down searches for Data Pool data, to view metadata and browse images online, and to convert and/or order Data Pool data.
4. Data Pool Order and Control GUI: a java-based web GUI which allows DAAC operations staff to track orders submitted through the Data Pool Web Access GUI.
5. Data Pool Maintenance GUI (EcDlDpm): a perl-based web GUI which allows DAAC operations staff to monitor Data Pool insert activity and to control the Data Pool configuration.
6. Data Pool Access Statistics utilities: perl utilities which parse firewall ftp logs (EcDlRollupFwFtpLogs.pl) and Data Pool Web Access custom code logs (EcDlRollupWebLogs.pl) for accesses to the Data Pool directories, and then roll up access information for storage in the Data Pool database.

7. Data Pool FTP Server: customized wu-ftp daemon, which supports ftp access to Data Pool directories and also provides a checksum-on-download service.
8. Data Pool Update Granule Expiration utility (EcDlUpdateGranule): a perl utility which allows operations staff to update the Data Pool expiration date and retention priority for specified Data Pool granules.
9. Data Pool Batch Insert Utility (EcDlBatchInsert.pl): a perl utility which allows operations staff to queue ECS or non-ECS data for insert into the Data Pool.
10. Data Pool Most Recent Insert Utility (EcDlMostRecentInsert.pl): a perl utility which creates files at the file system and data collection level of the Data Pool directory structure which contain information about granules recently inserted into the Data Pool at those levels.
11. Data Pool Collection Remapping Utility (EcDlRemap.pl): a perl utility which allows DAAC operations staff to remap all data in a Data Pool collection directory from one higher level collection group directory to another.
12. Data Pool Density Map Utility (EcDlDensityMapUtility.pl): a perl utility which calculates spatial density map information about Data Pool collections and stores this information in the Data Pool database. This utility normally runs as a cron job.
13. Data Pool Statistics Table Population Utility (EcDlPopulateStatTables.pl): a perl utility which populates tables in the Data Pool database which maintain counts of granules by collection and collection group, for use by the Web Access drill down GUI. This utility normally runs as a cron job.
14. HEG Front End: a java application which processes user input on the Data Pool Web Access GUI for orders requesting data conversion using the HDFEOS-to-GeoTiff (HEG) Converter Tool, creates parameter files to send to the HEG, and submits orders to the HEG.
15. Data Pool Database (DataPool[_<MODE>]): a Sybase database which stores Data Pool inventory and configuration information.

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 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.

5.2 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.
- There is no inter-DAAC planning at this time.

5.3 Infrastructure Subsystem

ASTER DAR Gateway CSCI

- 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 FTPPUSH for media type.

Subscription Server CSC/Subscription GUI

- The user can only subscribe to future granules.
- The Subscription Server validates String, long, double, date and time type qualifiers.

- Support for trigger persistence was merged in Release 5B.
- Subscription Server does not check security issues for updating its database tables. It assumes the client application does this.
- Under extreme rare situations, the Subscription Server might send two e-mail notifications for one subscription trigger.

Machine-to-Machine Gateway CSC

- Landsat-7 products may not be searched or ordered, since they require special order handling due to billing and floating scene subsetting. DAAC operations can specify which data types can be ordered via the SIPS interface. They must ensure the Landsat data types are excluded.
- The machine-to-machine gateway relies on ssh for the authentication of a connection.
- Each request must be associated with a MSS User ID. The operator has the choice of configuring a default MSS User ID for each MTMGW server. If there is no default for a server and the user does not provide a User ID in the request, the server rejects requests that do not specify a User ID.

5.4 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 Crack manually.
- The configuration files for TCP Wrappers can become difficult to manage if multiple versions exist. 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 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 is now 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.

5.5 Product Distribution System (PDS) Subsystem

- There is an inherent limitation on the system. When an order comes in it can be comprised of one Product Code (prod_code) or multiple product codes depending on the configuration of the system. For the purposes of the above receive_email and order_node values, these values are pulled from the first unit in the order and used for the remainder of the order.
- The PDSIS_Serverconfig_Tbl is used for storing the PDS job grouping per group data size or group size (number of units). The intent of this is to break large orders into smaller groups to pass to PDS. The configuration of the grouping depends on the grouping_config field.

If the grouping factor is "S" (granule sizes) then PDSIS sends data to ECS after that much data has been received from ECS. If the granule size totals equal the group_data_size those available units are sent to ECS.

If the grouping factor is "G" (Group limits) then PDSIS sends the data to ECS when the number of units currently staged by ECS is greater than or equal to group_unit_size. If the total number of unit is less than group_unit_size, then PDSIS sends the data to ECS only when all units are staged.

- Units statused as "X" (rejected) are ignored by the system
- When pdsis_serverconfig_tbl.server_mode = 'M' all groupings are ignored and, for all purposes, group_unit_size = 1 so data can be passed to PDS as it becomes available from ECS
- The grouping factor is ignored when current total data usage exceeds usage thresholds and the threshold release factor in PDSIS_Serverconfig_Tbl is "y". In such case, data is passed to PDS as it becomes available from ECS.

When units are statused as "R" (data staged by ECS) and the above grouping factors are met then data is inserted into the PDT_PDSINFO table. If an operator activates the job in PDS then only the records that are in the PDT_PDSINFO table are activated. Operators can activate partial orders if they want, however, for large MEDIA types such as DLT. This is a very inefficient usage of the media.

This page intentionally left blank.