

4.8 Common Services Tools

This section describes the tools used by DAAC operators on a day-to-day basis:

- 4.8.1 Common Desktop Environment (CDE) Tool
- 4.8.2 Firefox
- 4.8.3 Sun Java System Web Server
- 4.8.4 Batch Insert Utility
- 4.8.5 Update Granule
- 4.8.6 Data Pool Access Statistics Utility (DPASU) – Rollup Scripts
- 4.8.7 Data Pool Access Statistics Utility (DPASU) – Maintenance Scripts
- 4.8.8 Most Recent Data Pool Inserts Utility
- 4.8.9 Data Pool Collection-to-Group Remapping Utility
- 4.8.10 QA Update Utility
- 4.8.11 Data Pool Move Collections Utility
- 4.8.12 Data Pool Hidden Scrambler Utility
- 4.8.13 Data Pool Remove Collection Utility
- 4.8.14 Data Pool Band Backfill Utility
- 4.8.15 XML Replacement Utility
- 4.8.16 DPL cleanup orphan/phantom validation utility (EcDICleanupFilesOnDisk .pl)
- 4.8.17 DataPool Cleanup Granules Utility (EcDICleanupGranules.pl)
- 4.8.18 Link Checker Utility (EcDILinkCheck.pl)
- 4.8.19 AIM Tape Archive Consistency Checking Utility
- 4.8.20 EcDsCheckXMLArchive.pl
- 4.8.21 Cloud Cover Utility

This page intentionally left blank.

4.8.1 Common Desktop Environment (CDE) Tool

The ECS uses the Common Desktop Environment (CDE) Tool COTS package to manage X windows. It is a commercial graphical user interface for UNIX supporting AIX and Solaris operating systems. It provides users registered at an ECS site with generalized support for performing the basic operations listed in Table 4.8.1-1.

Table 4.8.1-1. Common ECS Operator Functions Performed with CDE

Operating Function	GUI	Description	When and Why to Use
Start a desktop session	Basic login with userid and password	Invokes the CDE window manager.	Access an ECS host.
Use the Front Panel	Front Panel window	Contains set of controls for performing common tasks (i.e., calendar, email, clock, print, file management).	As needed during work session.
Manage files	File Manager	File management tool.	Perform file navigation/manipulation.
Use Application Manager	Application Manager	How to run applications using Application Manager, the main repository for applications in CDE.	Need to invoke applications.
Customize the desktop environment	Style Manager	Allow for customizing the look and behavior of desktop.	Need to customize desktop environment.
Use text editor	Text Editor	Supports creation/editing of short documents (e.g., memos, mail, resource files).	Need to create short documents.
Print	Printing	Explains how to access printers.	Need to print/change default printer.
Use Terminal	Terminal	Explains how to display and customize terminal emulator windows on desktop.	Need to access control terminal window.
Use Icon editor	Icon Editor	Creates files for use as desktop icons or backdrops.	Need to create icons/backdrops.
Use Image Viewer	Image Viewer	Allows for capture, viewing, editing, printing, and translation of monochrome/color image files.	Need to perform image manipulation.

4.8.1.1 Quick Start Using the Common Desktop Environment (CDE) Tool

After being registered as an ECS user by the site administrator, the user accesses the CDE window manager by logging into an ECS host using a defined UserID and password.

4.8.1.2 CDE Main Screen

Figure 4.8.1-1 presents an example of the type of support provided by the CDE Window Manager.

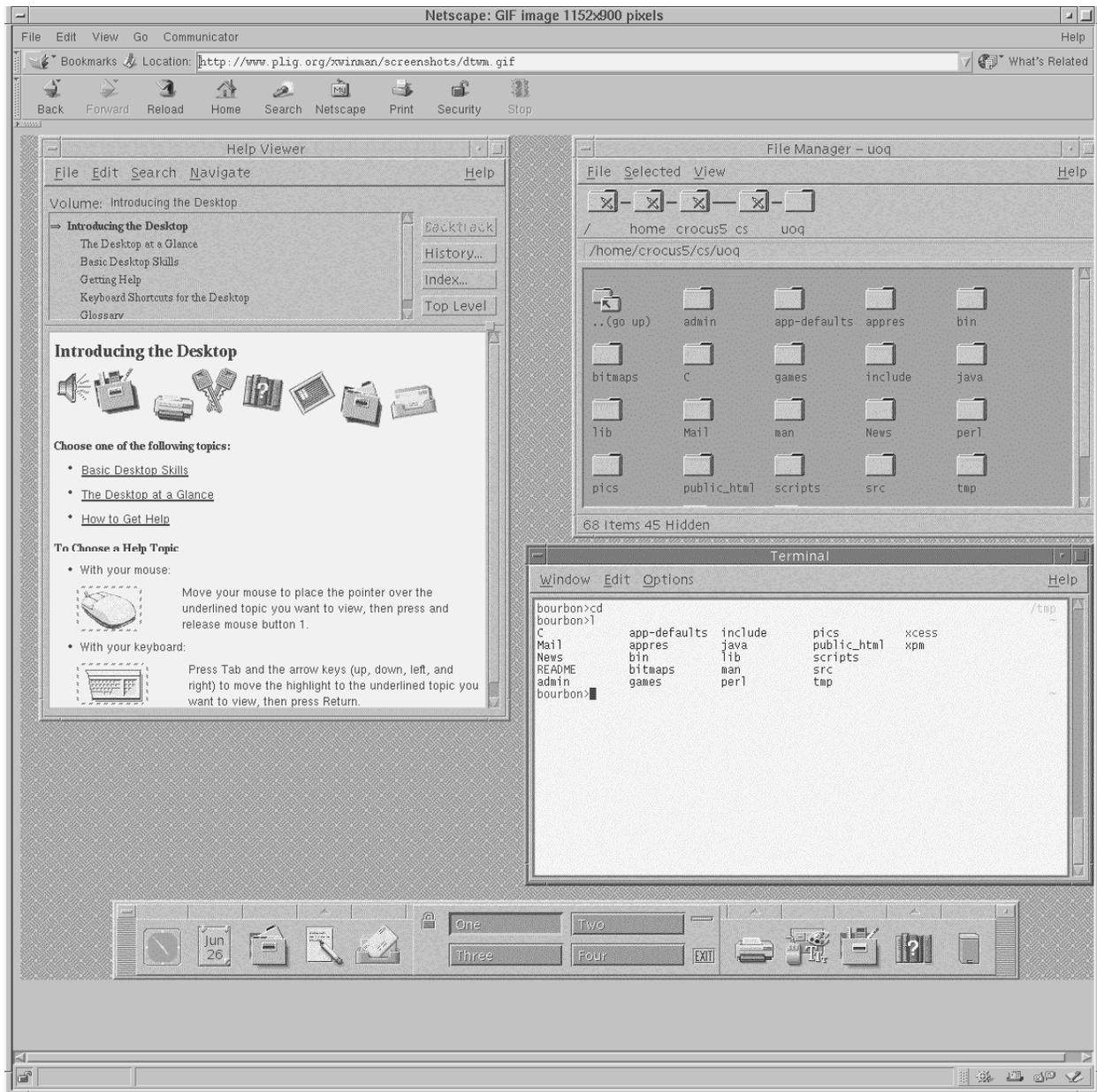


Figure 4.8.1-1. Example of CDE Window Manager Support Features

The Front Panel window at the lower part of the screen contains a set of icons allowing access to common support features. Through this panel the user can obtain time, date, monitor schedule,

access email, edit text files, print, access file manager to navigate the file system, and application manager to invoke and manage custom applications.

The Help Viewer window to the left of the screen is a support feature the user can invoke to obtain detailed online explanation of CDE support capabilities.

The File Manager window at the upper right of the screen supports navigating the file system and creating, deleting, and moving file objects.

The Terminal window below the File Manager on the screen allows UNIX command line access to operating system services.

In addition to the help accessible to the online user, detailed documentation of CDE capabilities from the user standpoint and the system administrator are available from the Sun vendor at the web location:

<http://docs.sun.com>.

4.8.1.3 Required Operating Environment

Refer to the Common Desktop Environment: Advanced User's and System Administrator's Guide.

4.8.1.4 Databases

None

4.8.1.5 Special Constraints

Access to CDE is available only to registered users of ECS sites.

4.8.1.6 Outputs

The Common Desktop only outputs event and error messages.

4.8.1.7 Event and Error Messages

CDE issues both status and error messages to the operator screen. Error messages are listed in the CDE support documentation accessible at the web link:

<http://docs.sun.com>.

4.8.1.8 Reports

None

This page intentionally left blank.

4.8.2 Firefox

Firefox is a GUI interface for browsing the World Wide Web (WWW) and for obtaining information from other sources. Some of the Firefox functions are:

- View and process text and html files as well as other MIME formats
- Read content of bulletin boards on the world-wide-web

Firefox is used to perform the operator functions listed in Table 4.8.2-1. Firefox's Help option offers additional information on functionality not explicitly mentioned here.

Table 4.8.2-1. Common ECS Operator Functions Performed with Firefox

Operating Function	Command/Action	Description	When and Why to Use
View Web Pages	Main window	<ul style="list-style-type: none">• Operator views pages written in HTML source code.• These pages provide images, text, and form templates.	To obtain information and to process user-interactive forms.
Process Forms	Main window	<ul style="list-style-type: none">• Forms are provided for operator input.• Certain operations require a password.	Used to search or manipulate the existing database (functions add, delete, modify.)
Browse Bulletin Boards (BB)	Main window	Allows for exchange of information with users and scientists that share the same interest.	To ask or provide information on the BB subject to a large community of users.

4.8.2.1 Quick Start Using Firefox

This section describes how to invoke Firefox. Upon startup, Firefox displays an initial web page that typically contains information, directions, and links to other web pages. A menu bar on the GUI provides access to a variety of Firefox features, including its on-line Help. To learn about Firefox's features, open the "Help" pulldown menu on the Firefox main screen and select "Help Contents". A web page appears that contains a search tool and links to various topics. The operator can select subjects in which he or she is interested by following the available links or by searching for additional web pages. By opening the "File" menu on the main page and selecting "Print", a hardcopy of the displayed text can be obtained.

4.8.2.1.1 Command Line Interface

To execute Firefox from the command line prompt use either:

> **`/tools/bin/firefox &`**

or

> **`/usr/ecs/OPS/COTS/firefox/firefox &`**

4.8.2.2 Firefox Main Screen

The first time invoked, Firefox displays its default home page, a startup screen like the one shown in Figure 4.8.2-1.

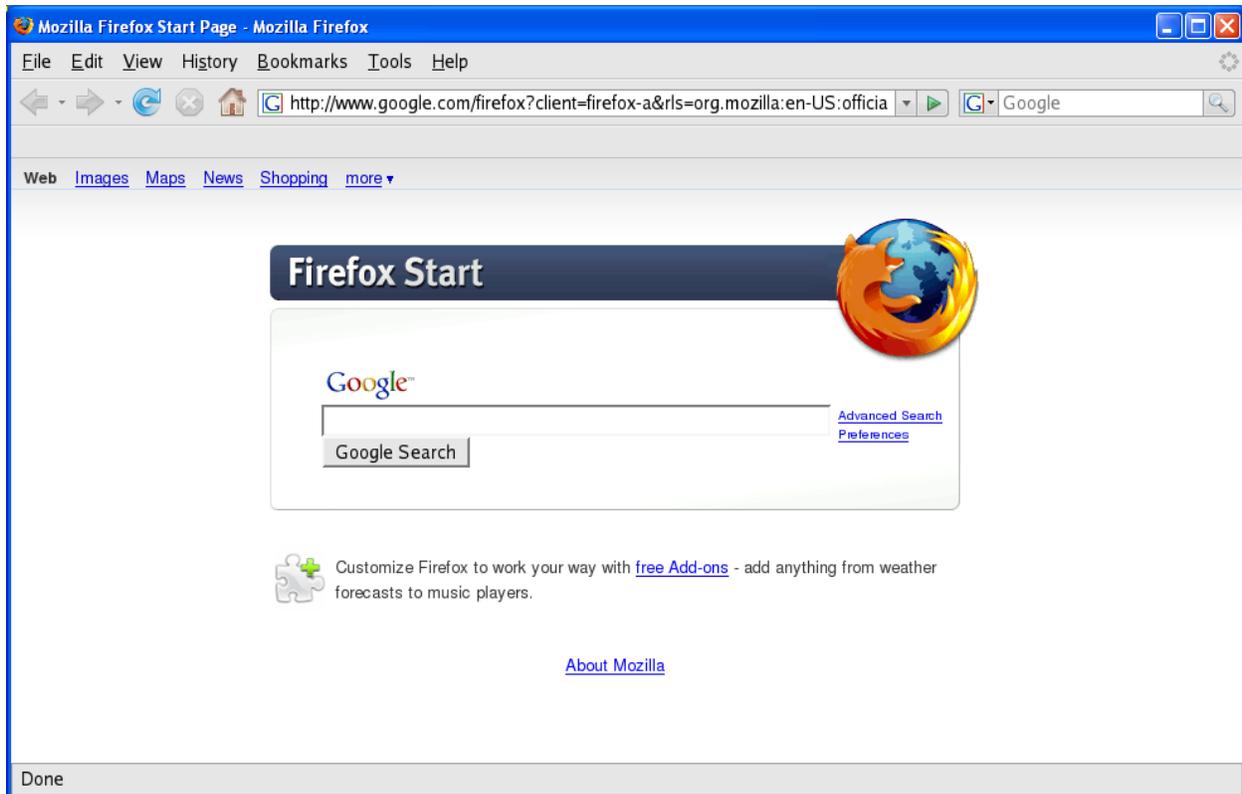


Figure 4.8.2-1. Firefox: Browser with Default Home Page

Operators can configure Firefox to use a different Home Page as the default. In addition, individuals can set a preferred Home Page by clicking **Tools** → **Options** on the menu bar, entering a URL in the Home Page field, then clicking **OK**. An example of a more useful Home Page is the ECS Data Handling System page shown in Figure 4.8.2-2.



Figure 4.8.2-2. Firefox: EDHS Home Page

From the start-up Firefox screen, the operator has several choices for loading pages in any of the MIME formats known by Firefox:

- Move the cursor to a link displayed on the page, and click on this link
- Enter or modify a URL displayed in the location bar (just below the menu bar).
- Click **View → Sidebar → Bookmarks** on the menu bar, and select a URL from the bookmarks sidebar that is displayed
- Click **File → Open File...** on the menu bar to browse for and open a file

It is recommended that operators create bookmarks of pages to be accessed frequently.

4.8.2.3 Required Operating Environment

For all COTS packages, appropriate information on operating environments, tunable parameters, environment variables, and a list of vendor documentation can be found in a CM-controlled document for each product. To find the documentation for Firefox, refer to the ECS Baseline

Information System web page. Contact your local CM Administrator for how to access this website at your site.

4.8.2.4 Databases

Firefox maintains a variety of information such as user preferences, security certificates, bookmarks, cookies, download history, extensions and add-ons. On Linux, these files are placed in the user's `~/mozilla/firefox/<profile>` directory.

4.8.2.5 Special Constraints

None

4.8.2.6 Outputs

Firefox provides the outputs listed in Table 4.8.2-2 below.

Table 4.8.2-2. Firefox Outputs

Output	Description and Format
Screen Display	Shows the Firefox browser GUI screen and adjusts to the screen format.
Hardcopy of Display Window	Printed version of the contents of the display window.
Display Window saved to disk	Contents of the display window can be saved to disk in Text, Source or Postscript format.
Modified, deleted or created data files	Processing of forms allows the operator to modify, delete or create data files.

4.8.2.7 Event and Error Messages

Firefox issues both status and error messages to document the status of loading a document or to display the reason for not loading a document.

4.8.2.8 Reports

None

4.8.3 Sun Java System Web Server

Sun Java System Web Server is a multi-process, multi-threaded, secure web server built on open standards. It provides high performance, reliability, scalability, and manageability for any size enterprise, and it includes modules for creating and managing Web content, for extending or replacing functions of the server (e.g., through Java servlets and JavaServer pages), and for providing application-specific services such as security and access control.

In EMD, Sun Java System Web Server is used by several subsystems to access HTML files and to service web-based applications. It is installed locally on machines that run EMD applications relying on it. A distinct instance of a Sun Java System Web Server is created for each such application, one per mode in which the application runs. For example, EMD's Order Manager, Data Pool GUI, and BMGT all need to use Sun Java System Web Server, and each of them runs in the three modes on sites' Data Pool Server machines. Consequently, nine instances of the Sun Java System Web server are required - one for each of the three applications in each mode. Applications communicate with the appropriate instance via a unique port number. The port numbers these Sun Web Servers use can be found in the EMD baseline document, 910-TDA-002, ECS Software Port Mapping Baseline.

An additional instance of the Sun Java System Web Server known as the Administration Server is created whenever Sun Java System Web Server is installed on a machine. You use it to manage all Web Server instances.

Table 4.8.3-1 summarizes the Sun Java System Web Server functions used by EMD and references vendor guides that describe their use. Release Notes are available on the Internet at <http://docs.sun.com/app/docs/coll/1653.3?l=en>.

Table 4.8.3-1. Common EMD Operator Functions Performed with the Sun Java System Web Server (1 of 3)

Operating Function	Command/Script	Description	When and Why to Use
Administer <i>Sun Java System</i> web servers	Configurations GUI	Allow operators to add and remove web server instances.	When applications needing web servers are installed or removed.
Set Administration Preferences	wadm (CLI) shell	Allow operators to: <ul style="list-style-type: none"> • Stop the Administration Server • Edit its listen socket settings • Change the user account under which its processes run • Change its superuser settings • Specify log file options, including log file rotation • Configure JRE paths 	When <i>Sun Java System</i> is installed and when the Administration Server needs reconfiguration.
Provide security and encrypt transactions	Configurations GUI	Allow operators to: <ul style="list-style-type: none"> • Create a trust database • Request, install, and manage VeriSign and other server certificates • Install and manage certificate revocation lists (CRLs) and compromised key lists (CKLs) • Enable client authentication 	As needed to activate security features designed to safeguard data, deny intruders access, and allow access to those authorized.

Table 4.8.3-1. Common EMD Operator Functions Performed with the Sun Java System Web Server (2 of 3)

Operating Function	Command/Script	Description	When and Why to Use
Configure web servers	Configurations GUI	Allows operators to: <ul style="list-style-type: none"> • Start and stop web server instances • Adjust performance settings • Edit configuration file (magnus.conf) settings and apply them to the server • Add and edit listen sockets • View, manage, and archive logs • Monitor server activity and quality of service • Edit file cache settings 	As needed to improve web server performance, troubleshoot problems, and support use by EMD custom code.
Analyze log files	Configurations GUI Virtual Servers GUI	Allows operators to: <ul style="list-style-type: none"> • View access logs • View error logs • Set logging preferences 	As needed to monitor and troubleshoot web server activities.
Monitor servers	Configurations GUI Virtual Servers GUI	Allows operators to: <ul style="list-style-type: none"> • Compile and view a variety of server performance statistics in real-time • Set bandwidth and max connections parameters for enforcing quality of service policies 	As needed to monitor, manage, and troubleshoot web server activities and to tune server performance.

Table 4.8.3-1. Common EMD Operator Functions Performed with the Sun Java System Web Server (3 of 3)

Operating Function	Command/Script	Description	When and Why to Use
Program the server	Virtual Servers GUI	Allows operators to: <ul style="list-style-type: none"> • Install CGI programs, Java Servlets and JavaServer Pages • Configure how the server is to run them 	When installing new server-side applications or changing how the applications are to be run.
Manage server content	Virtual Servers GUI	Allows operators to: <ul style="list-style-type: none"> • Set primary and additional document directories • Configure document preferences • Configure URL forwarding • Customize error responses • Specify a document footer • Restrict the use of file symbolic links • Set the server to parse HTML files • Set cache control directives 	When creating or altering web server instances, to specify where documents to be served are located. When customized responses to client requests are warranted. When restrictions are needed on information cached by proxy servers.

4.8.3.1 Quick Start Using Sun Java System Web Server

Sun Java System Web Servers are managed with the help of the following user interfaces:

- Administration Console (GUI).
- Command Line Interface (wadm shell).

You can use either the web based Administration Console or the wadm shell interface for creating and managing your web server instances. This document focuses on using the GUI-based Administration Console. Refer to the *Sun Java System Web Server 7.0 Update 3 CLI Reference Manual* for a detailed description of wadm shell commands, and to 914-TDA-428, *Sun Java System Web Server 7 Update 3 Maintenance Upgrade for Linux for the EOSDIS Core System (ECS)* for examples of how wadm can be used to create web server instances needed by ECS.

Note: EMD currently uses multiple instances of the web server rather than multiple virtual servers in a web server configuration. In past releases, Sun Java System Web Server’s virtual servers did not support unique configuration information.

The Administration Server must be running before the operator can access the Administration Console.

4.8.3.1.1 Command Line Interface

The preferred method for starting the Administration Server operationally is to type the following as root:

```
# /etc/init.d/webserver7-<xxxxxx> start
```

where <xxxxxx> is a code that was assigned during installation of the product.

This starts the Administration Server using the port specified during installation.

To start the Administration Console and proceed to access the functionality discussed in Table 4.8.3-1, start a web browser and then enter the URL for the Administration Server as follows:

```
http://<servername>.<ECSDomain>.<domain>:<portnumber>
```

The operator is then prompted for a username and a password. Once this information is entered the Administration Console web page appears as shown in Figure 4.8.3-1.

Note: The browser used for this task must be capable of supporting frames and JavaScript. Firefox 3.0, included in the EMD baseline, is capable of supporting both frames and Java Script.

4.8.3.2 Sun Java System Web Server Main Screen

The first page you see when you access the Administration Console is the Common Tasks page. This is Sun Java System's main web server administration screen.

Operators use the Common Tasks screen (Figure 4.8.3-1) to manage, add, remove, and migrate their web servers. It provides quick access to the more commonly-used screens accessible from other Administration Server pages. Selecting a configuration or virtual server and then clicking a listed task navigates the operator directly to the GUI or wizard that performs that task.

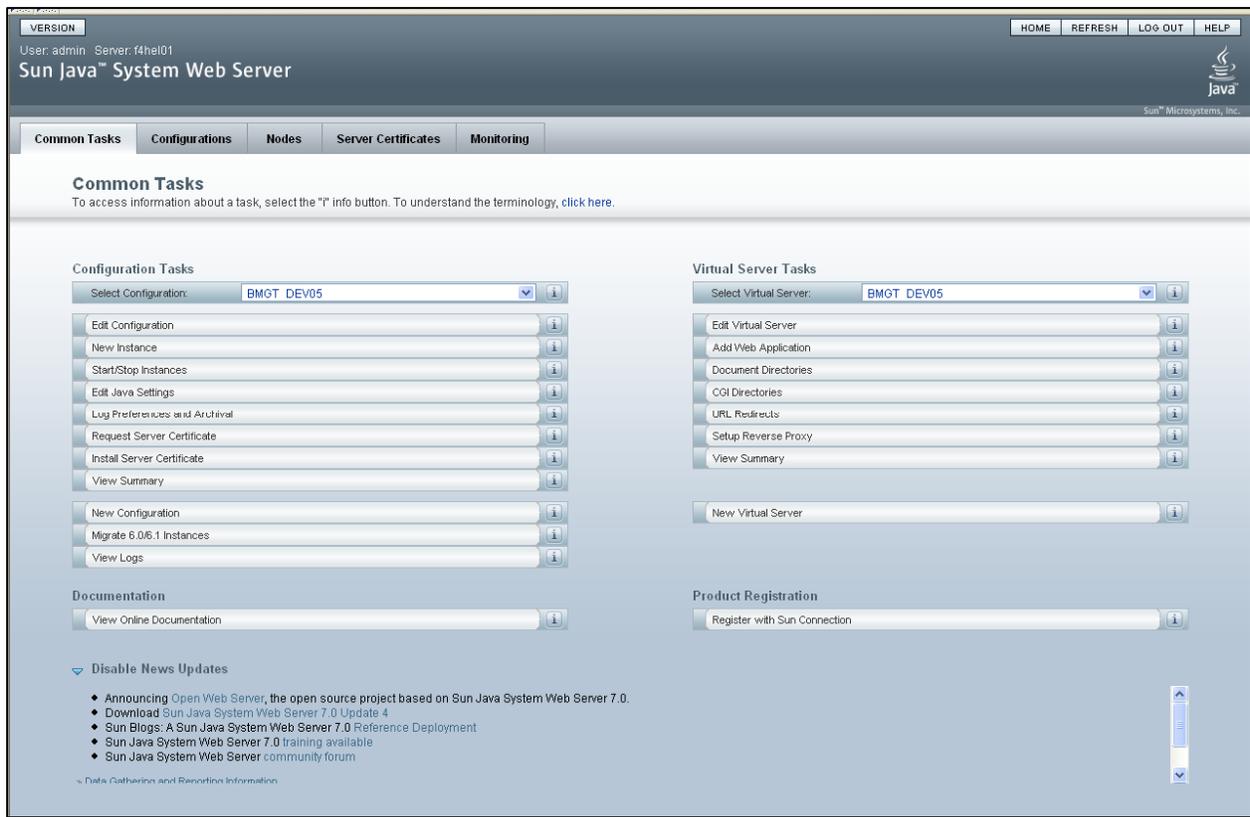


Figure 4.8.3-1. Sun Java System Web Server Common Tasks Screen

In addition, the screen has five tabs, each of which contains buttons for accessing Java forms to perform functions that govern the Administration Server or all the other web servers under its control. At this location (and on other tabbed pages), clicking any of the tabs may result in the appearance of child tabs. The actions provided by the child tabs are specific to the parent tab functionality.

The tabs are:

- Common Tasks tab – for adding (i.e., creating), duplicating, deploying, and deleting web server Operators invoke the Server Manager GUI by first selecting a web server from the tab's Select A Server pulldown menu, then pressing the Manage button
- Configurations tab – for adding (e.g., creating), duplicating, deploying, and deleting web server configurations. In order to start using a web server, you need to create a configuration. A configuration is a set of files and virtual servers that determine the data to be served. Click here to create a configuration. The wizard guides you through the process and creates a default virtual server and HTTP Listener for the configuration.
- Nodes tab – for configuring the server nodes and to add or modify server instances associated with the nodes. The Nodes page provides a high level overview of the configured nodes.

- Server Certificates tab – for requesting, installing, renewing, and deleting web server certificates. This page lists the server certificates available in all the configurations and lets you manage them.
- Monitoring tab – for viewing web server statistics, controlling and configuring web server monitoring, and viewing web server logs. Monitoring is done at both the configuration and web server instance levels. A Configurations sub-tab lists the deployed configurations along with some monitoring statistics at the configuration level. Click on the configuration name to view more monitoring details pertaining to that configuration. An Instances sub-tab lists all the instances deployed on all the nodes and general statistics for each. Click on an instance name to view its additional statistics.

The sections that follow describe two of the more commonly used screens.

4.8.3.2.1 Sun Java System Web Server: Configurations Screen

Operators use the Configurations GUI for managing their web server configurations. A configuration is a set of files and virtual servers that determine data that is to be served. The screen has nine buttons, six of which are active only when a check mark is placed in the box next to one of the listed configurations. The buttons are:

- New – for defining a new configuration. Clicking this tab launches a wizard for specifying a name, server name, document root, server user listener port and IP address, target nodes, and other optional information such as a CGI URI and path for the new web server.
- Deploy – for deploying the configuration on the target nodes. Clicking this tab launches a wizard that creates or updates appropriate files and directories on the hosts (nodes) on which the web server is to run. Before deploying, the wizard advises if changes have been made locally to configurations deployed previously and, if so, provides an option to upload (or “pull”) the configuration changes back to the Administration repository instead.
- Duplicate – for copying a server configuration to create a new one. Clicking this tab launches a wizard that requests a name for the new configuration.
- Start – for starting the configuration’s web server instance on all of its target nodes.
- Stop – for stopping the configuration’s web server instance on all of its target nodes.
- Restart – for re-starting the configuration’s web server instance on all of its target nodes.
- Delete – for removing a defined configuration and its web server instances from the system. Clicking this tab launches a wizard that lets you specify whether or not to retain the web server instances’ logs.
- Migrate – for migrating Sun Java System Web Servers from version 6 to version 7. This is not used in ECS.
- View logs – for gaining access to the events and errors logged by the web servers. Clicking this tab launches a wizard for identifying the log to view by type, node, configuration, and virtual server. An optional time range, event level, number of records, and number of records per page can be specified as well.

Clicking on any listed configuration invokes a GUI for editing the various characteristics for that configuration, such as its virtual servers (Section 4.8.3.2.2.).

Figure 4.8.3-2 shows the Sun Java System Web Server: Configurations screen.

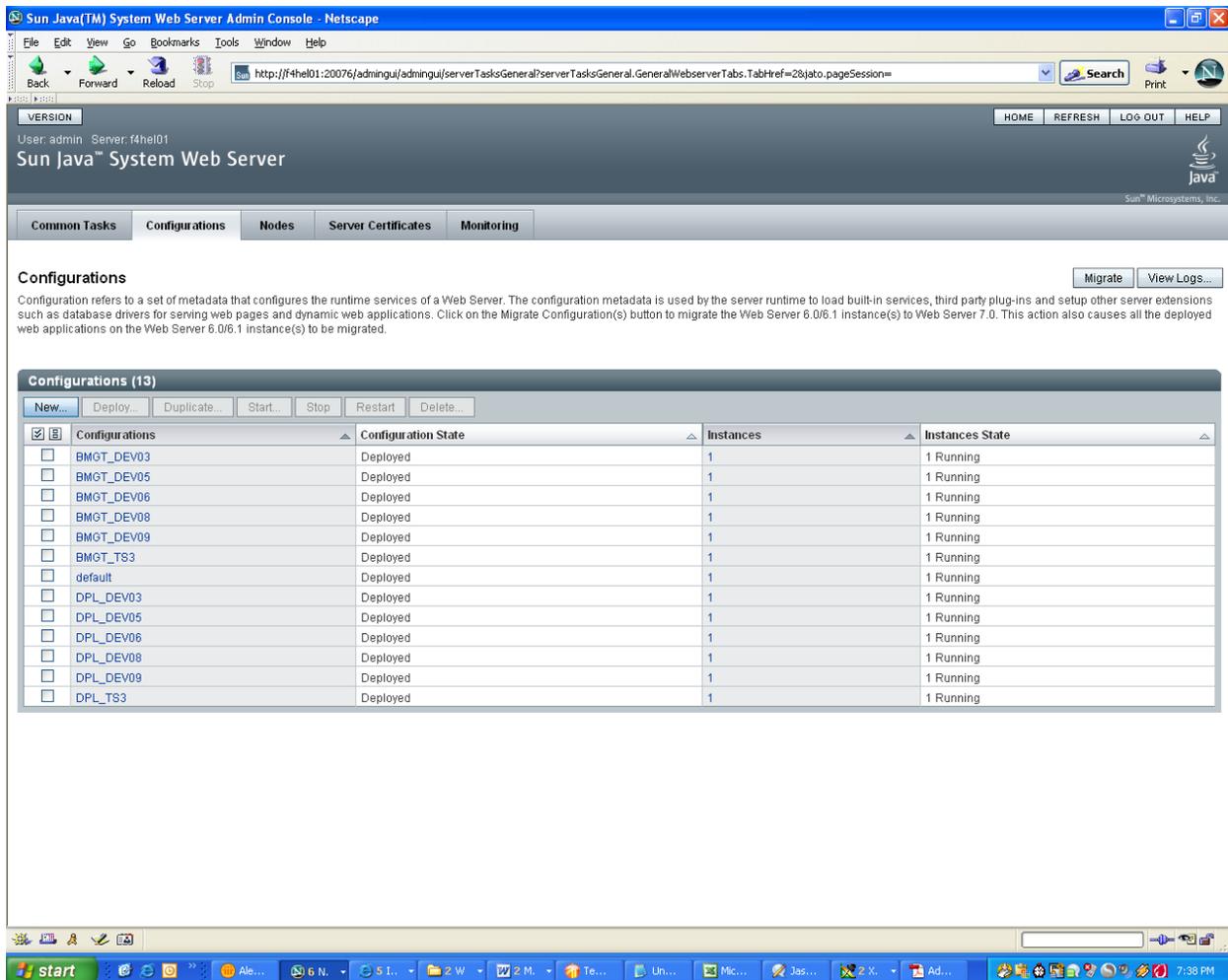


Figure 4.8.3-2. Sun Java System Web Server: Configurations Screen

4.8.3.2.2 Sun Java System Web Server Virtual Servers Screen

Operators use the Virtual Servers GUI to create and edit multiple virtual servers within a web server configuration. Virtual servers can have individual domain names, IP addresses, and some server monitoring capabilities. It is as though its users have their own web server. Each virtual server has an HTTP Listener specified. When a new request comes in, the server determines which virtual server to send it to, based on the configured HTTP Listener.

The screen has four buttons, three of which are active only when a check mark is placed in the box next to one of the listed configurations. The buttons are:

- **New** – for creating a new virtual server. Clicking this tab launches a wizard for specifying a name, the hosts, a document root, and an HTTP listener for the new virtual server.
- **Duplicate** – for copying a virtual server definition to create a new one. Clicking this tab launches a wizard that requests a name for the new virtual server.

- Add Web Application – for adding a web application archive (.war file) or web application path in the server. Clicking this tab launches a wizard for specifying the web application’s location, URI, and deployment directory, and whether to pre-compile the application’s JSPs to improve performance.
- Delete – for removing a defined configuration and its web server instances from the system. Clicking this tab launches a wizard that lets you specify whether or not to retain the web server instances’ logs.

Figure 4.8.3-3 shows the Sun Java System Web Server: Virtual Servers Screen.

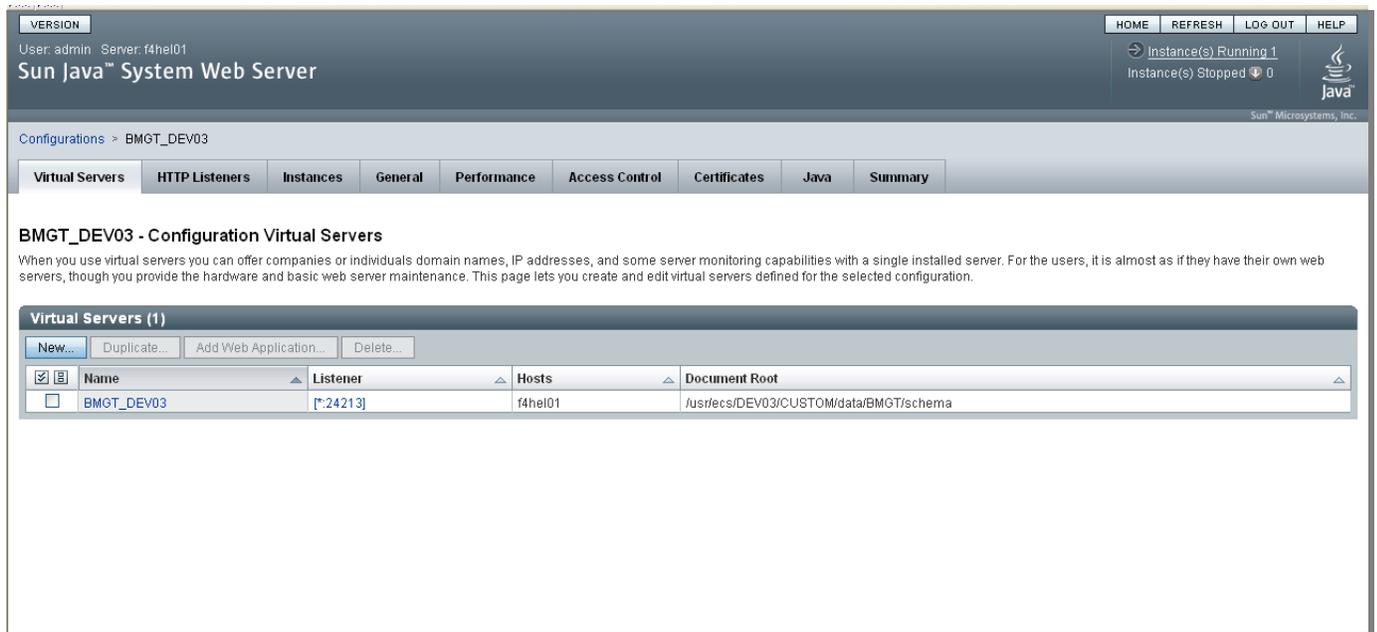


Figure 4.8.3-3. Sun Java System Web Server: Virtual Servers Screen

Other screens can be accessed via tabs on the Virtual Servers screen. The tabs include:

- HTTP Listeners tab – for adding and configuring HTTP Listeners. HTTP Listeners must have a unique combination of port number and IP address
- Instances tab – for managing the environment of a configuration’s web server daemon on a given node, and for starting and stopping the daemon..
- General tab – for editing the configuration’s properties such as log preferences, MIME types, monitoring settings, Web-based Distributed Authoring and Versioning (WebDAV), scheduled events, variables, and other characteristics. This tab also provides access to the Configuration Backups screen for restoring prior versions of the configuration.
- Performance – for adjusting the web server to improve performance. Tunable settings are available for thread pools, keep alives, DNS, SSL, file and ACL caches, CGI programs, and access log buffers.

- Access Control tab – for specifying who can access the administration server, which applications users can access, and who can access the web sites’ files and directories.
- Certificates tab – for requesting, installing, renewing and deleting server certificates.
- Java tab – for setting the path, class path and other properties for the Java virtual machine the web server is to use.
- Summary tab – for viewing and editing a large number of the settings for the configuration from a single page.

Clicking on any listed virtual server invokes a GUI for tailoring the characteristics of that virtual server, such as its document and CGI directories, HTTP listeners, variables, log preferences, web applications, search properties, and access controls.

4.8.3.3 Required Operating Environment

The Sun Java System Web Server is deployed to run on Linux machines. It can be administered via its web-based Administration Console or its command line interface (wadm shell). Operators have Firefox 3.5.4 to access the Administration Console, but they should have Java and cookies enabled in their browsers to use it.

4.8.3.4 Databases

The Sun Java System Web Server uses an internal, trust database to store public and private keys in support of Secure Socket Link encryption. The Administration Server and each server instance can have its own trust database. In addition, Sun Java System Web Server allows operators to define one or more Lightweight Directory Access Protocol (LDAP) databases that virtual servers can use for user authentication. The web servers themselves do not manage LDAP databases, and EMD does not currently use them. Refer to the *Sun Java System Web Server Administrator’s Guide* for further information on databases used by the Web Server.

4.8.3.5 Special Constraints

None

4.8.3.6 Outputs

The Web Server supports real-time monitoring of web servers’ activities. After enabling statistics, operators can view data about connections, the Domain Name Service (DNS), keep alives, cache, virtual servers, and more. These can help them identify how many resources their servers need.

Statistics are available at the configuration level, the server instance level, and the virtual server level. For a configuration, is reported about requests, errors, and response time. For server instances, data includes that as well as connection queues, keep alives, DNS, file cache, thread pools, and session threads. For virtual servers, data is available concerning requests, errors, connections, success and error responses, and web applications.

Statistics are activated by default on Web Server, but they can be disabled. To re-enable them, use the Monitoring Settings page, which is accessible from the General tab on the Virtual Servers GUI.

To monitor a greater variety of server statistics, use Sun Java System's *perfdump* utility. The utility must first be "installed" by editing the web server's *obj.conf* configuration file and restarting the server. Statistics can then be viewed by pointing a browser to <http://<host>/perf>. See the *Performance Tuning, Sizing, and Scaling Guide* for details.

4.8.3.7 Event and Error Messages

Each web server uses two files to record server activity. The *access* log file records requests to the server and server responses. The *error* log file lists errors the server has encountered. Both files typically reside in the web server's logs directory, but operators can control their location. Operators can also have the system automatically archive these files. See the *Administrator's Guide* for details.

For *access* logs only, operators can control the amount and format of what gets logged. They can specify whether to log accesses, what format to use, and whether the server should spend time looking up the domain names of the clients when they access a resource. They can also specify the file be written in common logfile format, flexible log format, or a user customizable format. Specify preferences using the Log Preferences page of the Server Manager GUI or edit the web server's configuration files directly. See the *Administrator's Guide* and the *NSAPI Developer's Guide* for details.

4.8.3.8 Reports

Operators can export performance data collected by the server into an XML or plain text report and publish it at any URI. (*/stats-xml* is the default). The report can contain a variety of statistics as described in Section 4.8.3.6 above. Results depend on the statistics that are collected, as determined by the web server's monitoring settings.

Operators can also run program *flexanlg*, the flexible log analyzer, from the command line. It is located in the *server-root/bin* directory. Its results depend on the logging that is done by the web servers. Refer to the *Sun Java System Web Server 6.1 Administrator's Guide*, Chapter 10 - "Using Log Files", for additional information.

This page intentionally left blank.

4.8.4 Batch Insert Utility

NOTE: This utility is being replaced with the “Publish Utility” (see section 4.7.17).

The Batch Insert Utility allows operators to insert granules residing in or outside of (non-ECS granules) the ECS archive into the Data Pool. It is a command line utility, which queues the granules up for dispatch by the Data Pool Action Driver (DPAD) for insertion by the Data Pool Insert Utility (DPIU). It accepts either a list of ECS granule identifiers or a list of non-ECS file names. A label identifying a batch of granules is specified as a command-line parameter so that operators can monitor a batch with the Data Pool Maintenance GUI. Thematic collections are also supported so the granules to be inserted can be linked to a theme.

Fault recovery capability is also supported, preventing inserts of duplicate actions inserted from a previous run.

Input is provided via input file or standard input.

4.8.4.1 Quick Start using the Batch Insert Utility

Enter the following command to start the Batch Insert Utility:

```
> EcDIBatchInsert.pl mode -ecs | -nonecs [ -file pathname ]  
    [ -granules id1,id2... ]  
    [ -collection collection ]  
    [ -actionsource B / R ]  
    [ -theme "theme_name" ] [ -label label ]  
    [ -rpriority priority ] [ -rperiod period ]  
    [ -dpriority priority ] [ -mdonly ]  
    [ -dplonly ] [ -verbose ]
```

Table 4.8.4-1 provides a description of these command line parameters.

Table 4.8.4-1. Command Line Parameters of the Batch Insert Utility (1 of 2)

Parameter Name	Description
<i>mode</i>	An input parameter specifying the mode of operation. This must be the first parameter passed, and it must be a valid, existing Data Pool mode with a format like OPS or TS1.
-ecs	Indicates that ECS granules are inserted. The input file (see -file) (or standard input) consists of a list of granule ids.
-nonecs	Indicates that non-ECS granules are inserted. The input file (see -file) (or standard input) consists of a list of XML file pathnames.
-file <i>pathname</i>	The pathname of the input file containing a list of either granule ids (if -ecs is specified) or XML pathnames (if -nonecs is specified).
-granules <i>id1,id2</i>	The list of granule ids specified on the command line.
-collection <i>collection</i>	The collection name for insert in the form of ShortName.VersionId.

Table 4.8.4-1. Command Line Parameters of the Batch Insert Utility (2 of 2)

Parameter Name	Description
-actionsource <i>B/R</i>	The action source specified to distinguish between publication and registration requests.
-dponly	Indicates the granules specified for inserts have to be present in data pool.
-theme " <i>theme_name</i> "	Theme name to be associated with granules. <i>theme_name</i> is a character string and must match an existing theme name in the Data Pool inventory. Enclose it in quotes if embedded blanks or other special characters are part of the name.
-label <i>label</i>	An identifying label to be linked to the batch of granules being inserted. <i>label</i> is a character string. If no batch label (-label) is supplied, the label is set to the first sixteen characters of the input filename (excluding the directory name). If standard input is used in lieu of an input file, a batch label must be specified with the -label option.
-rpriority <i>priority</i>	A retention priority to be applied to all granules being inserted. $255 \geq \textit{priority} \geq 1$
-rperiod <i>period</i>	Number of days to retain all granules being inserted in inventory.
-dpriority <i>priority</i>	A dispatch priority to be applied to all granules being inserted. $255 \geq \textit{priority} \geq 1$
-mdonly	Flag indicating only metadata files are inserted for all granules being inserted.
-verbose	Directs the utility to run using verbose option. Default is non-verbose.

Mandatory parameters include *mode* and either -ecs or -nonecs. *Mode* must be the first parameter supplied.

4.8.4.1.1 Batch Insert Utility Commands

Below are some examples for invoking this utility:

1. `EcDlBatchInsert.pl OPS -ecs -file /home/fred/ECSMODISgranules1 -verbose`

Adds actions to action insert queue for all ECS granules specified by granule ids in the input file. No -label parameter specified, so label is formed from first 16 characters of input filename (ECSMODISgranules). Runs in the verbose mode.

2. `cat /home/fred/ECSfile1 | EcDlBatchInsert.pl OPS -ecs -label MODIS_batch1 -verbose`

This example is similar to example 1 but using standard input instead of -file. Note that the -label parameter must be supplied since filename is not accessible to the utility.

3. `EcDlBatchInsert.pl OPS -nonecs -file /home/fred/nonECSVolcanogranules
-label Chig_volcano -theme "Chiginagak Volcano 2002"`

Adds actions to action insert queue for all non-ECS granules specified by XML pathnames in the input file. All granules are linked with theme name of "Chiginagak Volcano 2002" in inventory. Runs in the non-verbose mode.

4. `EcDlBatchInsert.pl OPS -ecs -file /home/fred/ECSMODISgranules1 -verbose -mdonly`

This example is similar to example 1 but only metadata files are inserted.

```
5. EcDlBatchInsert.pl OPS -ecs -file /home/fred/ECSMODISgranules1 -verbose -rpriority 200
```

This example is similar to example 1 with retention priority of granules to be set to 200 in the inventory.

```
6. EcDlBatchInsert.pl OPS -ecs -file /home/fred/ECSMODISgranules1 -verbose -rpriority 200 -rperiod 10 -dpriority 5
```

This example is similar to example 1 with retention priority of granules to be set to 200 in the inventory, retention period to last 10 days, and dispatch priority set to 5.

```
7. EcDlBatchInsert.pl OPS -ecs -granules 12345, 23456 -verbose
```

Adds actions to action insert queue for ECS granules specified on the command line: 12345 and 23456.

```
8. EcDlBatchInsert.pl OPS -ecs -granules 12345, 23456 -dplonly -verbose
```

This example is similar to example 7 with constraint of the specified granules have to be present in the data pool; no actions will be added to the insert queue if specified granules are not in the data pool.

```
9. EcDlBatchInsert.pl OPS -ecs -file /home/fred/ECSMODISgranules1 -actionsource R -verbose
```

This example is similar to example 1 with registration only option.

```
10. EcDlBatchInsert.pl OPS -ecs -collection AE_Land.001 -verbose
```

Adds actions to action insert queue for ECS granules from specified collection.

4.8.4.2 Batch Insert Utility Main Screen

The Batch Insert Utility does not have a main screen. It has a command line interface only.

4.8.4.3 Required Operating Environment

The Batch Insert Utility runs on Linux platforms.

4.8.4.4 Databases

Table 4.8.4-2 lists the supporting products this tool depends upon to function properly.

Table 4.8.4-2. Interface Protocols

Product Dependency	Protocols Used	Comments
Data Pool and AIM databases	SQL	Via SQL server machines
Perl DBI	DBD::Sybase	Requires proper install of base-lined version of Perl.

If a Sybase error occurs, you are most likely to see the actual Sybase error string displayed on the screen and in the log. Some errors can be the database server is unavailable, the connection to

the database was dropped, or there was an error executing the stored procedure. In the event of a Sybase-sourced error, the utility immediately stops running.

In the event that a connection to the Data Pool database or AIM database cannot be established, the utility may repeatedly attempt to connect to the database, depending on how the configuration file was set (see Section 4.8.4.4.1). If, for example, NUM_RETRIES was set to 5 and SLEEP_SEC was set to 10, this means it tries to connect 5 times, and waits 10 seconds before each attempt – a total of 50 seconds if all attempts are unsuccessful.

4.8.4.4.1 Configuration File Format – EcDIBatchInsert.CFG

The “config” file contains vital details about how to connect to the Sybase database. Without this file, the utility cannot run. The “config” file must be a single-entry plain text ASCII file, which has the following format:

```
SYB_USER = EcDIBatchInsert
SYB_SQL_SERVER = <string>
SYB_DBNAME = <string>
PGM_ID = <string>
NUM_RETRIES = <integer>
SLEEP_SEC = <integer>
```

Table 4.8.4-3 provides a description of each parameter included in the “config” file.

Table 4.8.4-3. Individual Parameter

Parameter Name	Description
SYB_USER	The user name for the Sybase connection.
SYB_SQL_SERVER	The name of the SQL server for this Sybase connection.
SYB_DBNAME	The name of the Data Pool database you intend to connect to
PGM_ID	Program ID used for connecting to the Data Pool database.
NUM_RETRIES	The number of times the utility attempts to connect to the database before exiting. The recommended default is 5.
SLEEP_SEC	The number of seconds the utility waits ('sleep') between connection attempts. The recommended default is 10.

4.8.4.5 Special Constraints

The Batch Insert Utility runs only if the Data Pool and AIM database servers are running and if the databases are available. It also assumes the stored procedures are present.

4.8.4.6 Outputs

Output of events and errors is always appended to a single log file.

4.8.4.7 Event and Error Messages

Events and error messages are written to the log file. A usage message is displayed to the screen when command-line parameters are incorrectly specified.

The utility produces a log file called EcDIBatchInsert.log in the /usr/ecs/<mode>/CUSTOM/logs directory. If this log file already exists, the new information is automatically appended. If there is no existing log file by this name, a new log file with this name is automatically created.

Since the log file may grow to a considerable size after constant use, it is recommended that it be saved off into a separate file from time to time for maintainability.

4.8.4.8 Reports

None

This page intentionally left blank.

4.8.5 Update Granule

Deleted. Not applicable for Release 7.23.

This page intentionally left blank.

4.8.6 Data Pool Access Statistics Utility (DPASU) – Rollup Scripts

The Data Pool Access Statistics Utility (hereafter referred to as “DPASU”) provides the ECS Operations Staff with several capabilities related to collecting access statistics for the Data Pool database. The DPASU encompasses two types of scripts: rollup and maintenance. The rollup scripts read and parse access logs to compile statistics and store those records in the Data Pool database, while the maintenance scripts backup, restore, and delete data in the related Data Pool database tables.

These scripts may be run on the command-line, and must be run with an operations mode. Details and instructions on how to run and configure these scripts are provided in subsequent sections.

4.8.6.1 Data Pool Access Rollup Scripts

The Data Pool access rollup scripts provide the ECS Operations Staff with the capability to parse the Data Pool web access and FTP logs for Data Pool access information and store the access information in the Data Pool database. For each Data Pool file access found in the FTP or web access logs, the rollup scripts store into the Data Pool database (in the DIGranuleAccess table) the time of access, the corresponding granule ID, the file type (metadata, browse or science), the file size (in bytes), the access type (FTP or HTTP), and the age of granule at access time (i.e., the number of days the granule has been in the Data Pool at the time of access). Such information collected over a period of time can provide useful statistical information regarding the Data Pool access patterns and provides insight into planning future support.

There are three Data Pool access rollup scripts: one for rolling up accesses to the Data Pool via the Web Access GUI (EcDIRollupWebLogs.pl), and the other two for rolling up accesses to the Data Pool via anonymous ftp. The DAAC chooses which of the two ftp rollup scripts to use depending on the kind of firewall in place at that DAAC. The EcDIRollupFwFtpLogs.pl script is used at DAACs with an EMD-supplied PORTUS firewall; the EcDIRollupWuFtpLogs.pl script is used at DAACs with any other kind of firewall (e.g., at ASDC). All three rollup scripts are installed and run on the Data Pool host x4dpl01.

For DAACs with a PORTUS firewall, an additional EMD-supplied script runs once per day on the firewall host. This script produces a subset of the firewall log containing only ftp accesses to the x4dpl01 host. This subset of the firewall log is transferred to the `usr/ecs/OPS/COTS/firewall/logs` directory on the x4dpl01 host, with a file name of `datapoolftplog.<n>`, where `<n>` is an integer from 0 to 6, representing the day of the week when the log extract process was run. The EcDIRollupFwFtpLogs.pl script on the x4dpl01 host parses this `datapoolftplog.<n>` file.

For DAACs which do not have use a PORTUS firewall, the EcDIRollupWuFtplogs.pl parses the `wu-ftp` log on the x4dpl01 host (`/var/adm/xferlog`).

Each rollup script is a command-line utility allowing the operator to optionally pass input parameters. Operationally, the rollup script is run in a cron job, with the crontab file specifying when the rollup script shall start its daily execution. Each time the rollup scripts are run, they roll

up the Data Pool accesses that occurred over a specified 24-hour period in the past and store them into the Data Pool database.

By default, the start date of the 24-hour rollup period is one day prior to the date when the rollup script is executed. However, the rollup scripts allow a non-default start date of the rollup period to be specified via command line. This capability is provided to allow the DAAC operator to run the rollup scripts manually to compile statistics for a particular date for which the regular cron job may not have run for some reason.

To allow flexibility for each DAAC to specify the start time of the 24-hour rollup period, the start time is provided as a configuration parameter (ROLLUP_START_TIME). (See Section 4.8.6.3.3 for description of configuration parameters.) For example, if the start time is configured as 0:00 hours (midnight), then the rollup period will always cover a 24 hour period starting from 0:00 on the rollup start date. If the start time is configured as 6:00 hours, the rollup period always covers a 24-hour period starting from 6AM of the rollup start date.

Each of the Data Pool access rollup scripts work as follows. The script first parses the specified FTP or Web access log(s) for Data Pool access events. After the log files are parsed, the captured data is written to a temporary “flat file” – a tab-delimited text file. This file gets exported to the Data Pool database, where it is stored in a temporary table (DIftpAccessLog or DIWebAccessLog). The rollup script then uses information in the temporary access log tables and other Data Pool database tables to determine the information to be written to the DIGranuleAccess table. The flat file is removed and an entry is made into the DIAccessRollup table to keep a record of which periods have been successfully “rolled up” to prevent the accidental reprocessing of that period.

Normally the Data Pool access rollup scripts are run by cron. Unless the cron job was completely successful, no entry is made into the DIAccessRollup record table to indicate the rollup period was processed. Therefore, the DAAC operator is able to reprocess that period by manually running the rollup scripts from the command line.

4.8.6.1.1 Invoking the Data Pool Access Rollup Scripts from the Command Line Interface

Entering the following commands run the rollup scripts:

% EcDIRollupWebLogs.pl <command line parameters>

% EcDIRollupFwFtpLogs.pl <command line parameters>

% EcDIRollupWuFtpLogs.pl <command line parameters>

There are various optional and required command line parameters used in combination with each other. Table 4.8.6-1 provides a description of these parameters.

Table 4.8.6-1. Command Line Parameters of the DPASU

Parameter Name	Necessity	Description
<MODE>	Required	Indicates the Data Pool MODE the script is to run in. This parameter has the following constraints: It must be the first parameter with no label The <MODE> must imply a valid directory path An example of a <MODE> is OPS, TS1, TS2, et cetera.
-noprompt	Optional	Turns on the “noprompt” display mode – suppressing all output to the screen. This should be used in cron jobs or other scenarios where output to a display is not desired. The default display mode writes messages to the screen.
-flatfile <path/file>	Optional	Provides an alternative path/file name for the flat file produced by the parser. This is only useful with the –nodelete option.
-nodelete	Optional	Prevents the flat file from being deleted once the DPASU completes its run.
-fwftp <path/file(s)>	Optional	Indicates an alternative FTP log path/file(s) to be used instead of the configured default path/file(s). Use of a wildcard character is permitted in the file name, but if a wildcard is used, the path/file name must be enclosed in quotes (e.g. “/usr/ecs/OPS/COTS/firewall/logs/datapoolftplog.*” or “/home/allmode/archive/xferlog.0”). For EcDIRollupFwFtpLogs.pl or EcDIRollupWuFtpLogs.pl only.
-web <path/file(s)>	Optional	Indicates an alternative web log path/file(s) to be used instead of the configured default path/file. Use of a wildcard character is permitted in the file name, but if a wildcard is used, the path/file name must be enclosed in quotes (e.g., “usr/ecs/OPS/CUSTOM/logs/WebAccess.log”). For EcDIRollupWebLogs.pl only
-start <date>	Optional	Indicates the alternative start date for the rollup period with the format YYYY/MM/DD. This can be used to process previously uncovered periods. The default date is the date cron actually runs the DPASU on a day-to-day basis.

4.8.6.1.2 Default Rollup Period

By default, the rollup period begins 24 hours before the current date (i.e., the date on which the rollup script is run, either manually or by cron) plus the configured rollup start time (see Section 4.8.6.3.3 for configuration of the ROLLUP_START_TIME parameter). For example, if the rollup script is run on September 23, and the configured rollup start time is “2:00”, the Rollup period begins on September 22 at 2:00 a.m. and ends on September 23 at 1:59 a.m.

This means the Rollup script scans the specified log(s) for all entries having an access time between Sep 22 2:00 a.m. and Sep 23 1:59 a.m. Note that the 24-hour rollup period must be in the past as compared to the time the rollup script is run. The rollup script does not execute and terminates with an error if it detects the rollup period that was specified spans into a future time.

4.8.6.1.3 Specifying an Optional Start Date of Rollup Using `–start` Option

The rollup scripts allow an optional rollup period start date to be specified via command line using the `–start` option. With this option, a valid date must be entered in the following ordinal format:

YYYY/MM/DD

Because this is an optional *start* date, the end date gets 24 hours *added* to it. Therefore, if at the command-line the operator enters `–start 2002/09/18` (presumably because cron failed to run the DPASU Rollup on that day), the rollup scripts look for all entries with access times between 09/18/2002 02:00 and 09/19/2002 01:59.

The scripts check the input date at the command line to make sure it is valid before it parses the logs.

4.8.6.1.4 Running Data Pool Access Rollup Scripts with *cron*

The Data Pool access rollup scripts are run by cron on a daily basis at a consistent time of day. There are a number of factors to consider when determining at what time to run the rollup scripts each day. Factors are:

- a. The rollup scripts should be run at a time of day that is **AFTER** the configured rollup start time. (A good rule of thumb is to have the scripts run at least one half hour after the rollup start time.) This ensures the 24-hour rollup period has completed at the time the rollup scripts are run.

Example 1: If the rollup start time is 2:00 a.m., the cron should run the rollup scripts at a time after 2:30 a.m.

Example 2: If the rollup start time is 22:00, the cron should run the rollup scripts at a time after 22:30, but *not* after 23:59 because any time after that is the next day.

- b. It is recommended the rollup scripts be run by cron at a time of day when Data Pool access activity is low – e.g., during the early morning hours.
- c. The rollup scripts should be run **BEFORE** the daily Data Pool Cleanup script is run, to minimize chances that information about files accessed during the 24-hour rollup period has been removed from the Data Pool database. (If this information has been removed, the rollup scripts are unable to write information for those files in the `DIGranuleAccess` table.)
- d. The rotation/renaming times of the Web Access and FIREWALL FTP log files and the time the corresponding rollup script is run must be taken into consideration in determining, which log files to parse and whether to use a wildcard in the specification of the log file path.

For example, consider the case where the FIREWALL FTP log is rotated/renamed each day at 01:00, and the FTP rollup script is run at 03:00 with a rollup start time of 02:00. When the rollup script is run at 03:00 on September 22, 2002, the rollup period is

September 21, 2002 02:00 through September 22, 2002 01:59. The FIREWALL FTP log (e.g. datapoolftplog.1) which was rotated/renamed at 01:00, now only contains accesses for the time period September 22, 2002 01:00 through September 22, 2002 03:00 (the current time). The previous FIREWALL FTP log (e.g. datapoolftplog.0), contains accesses for the time period September 21, 2002 01:00 through September 22, 2002 00:59. To capture information for the entire rollup period, the ftp rollup script must be configured to parse both the datapoolftplog.1 and datapoolftplog.0.

This may be accomplished either by running the ftp rollup script twice, once against datapoolftplog.1 and once against datapoolftplog.0, or by running the script once and using a wildcard to specify the ftp log path. (Note that wildcard path names must be enclosed in quotes if used on the command line with the `-web` or `-fwftp` command line parameters, but do NOT need to be enclosed in quotes if used with the configuration parameters `WEB_LOG_PATH` or `FTP_FIREWALL_LOG_PATH`. See sections 4.8.6.1.5 and 4.8.6.1.6)

- e. To prevent or minimize the chances of database contention, it is recommended the daily cron job for rolling up FIREWALL FTP access logs and the daily cron job for rolling up web access logs be staggered, so the two rollup scripts do not run at the same time.

In the case that *cron* fails to run the Data Pool access rollup scripts on a given day, the operator can manually run either script, specifying the date(s) missed using the `-start` command line parameter.

4.8.6.1.5 Specifying Alternative Paths for FIREWALL FTP or Web Access Logs

The operator can specify an access log file path different than that specified in the configuration file by using the `-web` and/or `-fwftp` command line parameters. If alternative access log file paths are used with the command line options `-web` and `-fwftp`, any wildcards used to indicate multiple files matching a pattern need to be enclosed in quotes. If they are not, the rollup scripts cannot use the files you intended. The scripts *internally* (i.e., not the shell) match all files indicated by wildcards on the command line.

For example, the path

```
/usr/ecs/OPS/COTS/firewall/logs/datapoolftplog.*
```

must be enclosed in quotes as follows

```
"/usr/ecs/OPS/COTS/firewall/logs/datapoolftplog.*"
```

to ensure the wildcard character (*) is properly passed

Keep in mind that quotes around wildcard path names are only required on the command line; they are NOT required in the configuration file.

4.8.6.1.6 Intermediate Flat File

The rollup scripts create an intermediate flat file from the log entries that contain all the data that will be exported to the database via bulk copy procedure (bcp). Normally, this file is temporarily

placed in a data directory and then deleted, once the scripts have completed running. The operator can keep that flat file by specifying the **-nodelete** option. By default, the intermediate flat file is created in the following directory:

```
usr/ecs/<MODE>/CUSTOM/data/DPL/
```

The operator can specify an alternate path and name for this file on the command line using the **-flatfile** option.

4.8.6.1.7 Command-line Examples

Here are some examples of executing the Data Pool access rollup scripts from the command line.

Example 1:

```
EcDIRollupWebLogs.pl OPS -noprompt -nodelete -start 2002/12/22
```

Run Web rollup script in -noprompt display mode for an optional 24-hour rollup period starting from December 22, 2002, at the configured rollup start time. The -nodelete option prevents the flat file from being erased upon completion.

Example 2:

```
EcDIRollupFwFtpLogs.pl OPS -noprompt
```

```
EcDIRollupWuFtpLogs.pl OPS -noprompt
```

Run FIREWALL FTP / wuftp rollup script in -noprompt display mode for the default 24-hour rollup period starting from yesterday at the configured rollup start time. This example is typical of syntax used in the crontab file.

Example 3

```
EcDIRollupFwFtpLogs.pl OPS -start 2002/02/15 -fwftp "/usr/logs/*.log "
```

```
EcDIRollupWuFtpLogs.pl OPS -start 2002/02/15 -fwftp  
"/home/allmode/archive/xferlog.0"
```

Run FIREWALL FTP Rollup script in prompted mode, for an optional 24-hour rollup period starting from February 15, 2002, at the configured rollup start time, but use the FIREWALL FTP access logs stored in an alternative path /usr/logs.

4.8.6.2 Data Pool Access Statistics Main Screen

The Data Pool Access Statistics does not have a main screen. It is a command line interface.

4.8.6.3 Required Operating Environment

The Data Pool access rollup scripts run in a Linux operating environment.

4.8.6.3.1 Interfaces to supporting products

Table 4.8.6-2 lists the supporting products that this tool depends upon in order to function properly.

Table 4.8.6-2. Interface Protocols

Product Dependency	Protocols Used	Comments
Data Pool database	SQL	Via SQL server machines
Perl DBI	DBD::Sybase	Requires proper install of Perl

4.8.6.3.2 Access Log File Formats

The Data Pool access rollup scripts are dependent on a particular format of both the FIREWALL FTP and Web access logs. If the format of these log files changes, it is quite possible the scripts can incorrectly read certain fields and consequently provide incorrect or misleading rollup reports, and can even prevent the scripts from running at all. It is important to have the rollup script code adjusted if the FIREWALL FTP or WEB access log formats change in any way. Subsequent sections provide format details for these access logs.

4.8.6.3.3 Configuration Files for Data Pool Access Rollup Scripts

The Data Pool access rollup scripts use configuration files containing details about how to connect to Sybase and about where the log files exist. The file *EcDIWebRollup.CFG* contains the configuration parameters for the Web Rollup script. The file *EcDIftpFwRollup.CFG* contains the configuration parameters for the Firewall FTP Rollup script. The file *EcDIWuFtpRollup.CFG* contains the configuration parameters for the wu-ftp Rollup script. Without the configuration files, the scripts can not run. Each configuration file must be a plain text ASCII file, which has the following format, not necessarily in this order:

```
SYB_USER = <string>
SYB_SQL_SERVER = <string>
SYB_DBNAME = <string>
NUM_RETRIES = <integer>
SLEEP_SEC = <integer>
WEB_LOG_PATH = <path and file name>
FTP_FIREWALL_LOG_PATH = <path and file name> (for EcDIRollupFwFtpLogs.pl)
FTP_LOG_PATH = <path and file name> (for EcDIRollupWuFtpLogs.pl)
ROLLUP_START_TIME = <time of day>
```

Table 4.8.6-3 describes the individual configuration parameters mentioned above.

Table 4.8.6-3. Data Pool Access Configuration Parameters for Rollup Scripts

Parameter Name	Description
SYB_USER	The user name for the Sybase connection.
SYB_SQL_SERVER	The name of the SQL server for this Sybase connection.
SYB_DBNAME	The name of the Data Pool database you intend to connect to.
NUM_RETRIES	The number of times the utility attempts to connect to the database. The recommended default is 5.
SLEEP_SEC	The number of seconds the utility waits ('sleep') between connection attempts. The recommended default is 10.
WEB_LOG_PATH	The path and file name for the Data Pool Web Access custom code log. This parameter is valid for EcDIWebRollup.CFG only. The web rollup script automatically uses this path (and file or files) if an alternative one is not explicitly provided. Wildcards are permitted and do not need to be enclosed in quotes.
FTP_FIREWALL_LOG_PATH	The path and file name for the default ftp access log. This parameter is valid for EcDIFwFtpRollup.CFG only. The FIREWALL FTP rollup script automatically uses this path (and file or files) unless an alternative one is explicitly provided using the –fwftp command line parameter. Wildcards are permitted and do not need to be enclosed in quotes.
FTP_LOG_PATH	The path and file name for the default wu-ftp access log. This parameter is valid for EcDIWuFtpRollup.CFG only. The rollup script for the wu-ftp log automatically uses this path (and file or files) unless an alternative one is explicitly provided using the –fwftp command line parameter.
ROLLUP_START_TIME	The configurable time of day the rollup script uses as the start time of the 24 hour rollup period. 24-hour time must be used for this entry, e.g., "3:00" or "18:00".

4.8.6.4 Databases

The Data Pool Access Statistics utility uses the Sybase ASE Server.

4.8.6.5 Special Constraints/Dependencies

The Data Pool access rollup scripts function only if the Data Pool database server is running and if the Data Pool database is available. The rollup scripts also assume the required stored procedures are present in the Data Pool database.

Special modules are also required to run this utility. If those modules are not present or are located in an unfamiliar directory, it fails to run. Table 4.8.6-4 describes the modules required to run the rollup scripts.

Table 4.8.6-4. Data Pool Access Special Modules

Name	Description
EcDIDbInterface.pm	Database interface and connection module.
EcDIDateTime.pm	Date/time grabber with millisecond resolution.

4.8.6.6 Outputs

Rollup information is entered in the Data Pool database in the DIGranuleAccess table. If the -noprompt option is not on, examining status and other messages are printed to the screen. Log messages are also recorded (see below).

4.8.6.7 Event and Error Messages

All event and error messages generated from the rollup scripts are written to the respective log files. When the scripts are run in the prompted mode (default), the messages are also displayed to the screen in addition to writing to the logs.

4.8.6.8 Reports

None.

4.8.6.9 Recovery Procedures

In the case that *cron* fails to run the Data Pool access rollup scripts on a given day, the operator may manually run either script, specifying the date(s) missed using the -start command line parameter. See Section 4.8.6.1.3 for details.

This page intentionally left blank.

4.8.7 Data Pool Access Statistics Utility (DPASU) – Maintenance Scripts

The Data Pool Access Statistics Utility (hereafter referred to as “DPASU”) provides the ECS Operations Staff with several capabilities related to collecting access statistics for the Data Pool database. The DPASU encompasses two types of scripts: rollup and maintenance. The rollup scripts read and parse access logs to compile statistics and store those records in the Data Pool database, while the maintenance scripts backup, restore, and delete data in the related Data Pool database tables.

These scripts may be run on the command-line, and must be run with an operations mode. Details and instructions on how to run and configure these scripts are provided in subsequent sections.

4.8.7.1 Data Pool Access Maintenance Scripts

The Data Pool access maintenance scripts are operational support tools used for archiving, deleting, and backing up granule access data in the Data Pool database. Each of these scripts can be run on the command line and connects to the Data Pool database to process data contained therein. These scripts are installed and run on the Data Pool database host (x0acg0n), in the /usr/ecs/<mode>/CUSTOM/dbms/DPL directory. All of these scripts involve access to the Data Pool tables DIGranuleAccess, DIGranuleSubscription, and DIAccessRollup.

Archive Utility - *DI DbArchiveAccessStat*

This script archives data contained in DIGranuleAccess, DIGranuleSubscription, and DIAccessRollup by writing this data to an ASCII file based on an operator-specified time range.

Delete Utility - *DI DbDeleteAccessStat*

This script removes data contained in DIGranuleAccess, DIGranuleSubscription, and DIAccessRollup based on an operator-specified time range.

Restore Utility - *DI DbRestoreAccessStat*

This script restores data archived by the archive utility (contained in the ASCII file) into DIGranuleAccess, DIGranuleSubscription, and DIAccessRollup.

4.8.7.2 Invoking the Maintenance Utilities from the Command Line Interface

Entering the following commands start the maintenance utilities:

> **UtilityName <command line parameters>**

There are various command line parameters used in combination with each other. Table 4.8.7-1 provides a description of these parameters

Table 4.8.7-1. Command Line Parameters of the DPASU Access Maintenance Scripts

Parameter Name	Description
<MODE>	The mode in which the utility is being executed.
<STARTDATE>	The beginning date time range for archiving, deleting, or restoring the data. The format for this parameter is yyyyymmdd.
<STOPDATE>	The ending date time range for archiving, deleting, or restoring the data. The format for this parameter is yyyyymmdd.
<ARCHIVEDIR>	The absolute path where the generated ASCII files are stored when archiving or restoring data (this parameter only applies to the archiving and restoring scripts). The ASCII files are generated from the archive utility. The file name follows the convention <tablename>.dat.<startdate><stopdate>.
<USERNAME>	The Sybase login name.
<SERVER>	The Sybase Server where the Data Pool database located.
<DBNAME>	The name of the Data Pool database.

The parameters shown here are those used for all of the maintenance scripts. See the “Utility Commands” section for each script for specific usage. Please note that these parameters must be provided in the exact order as shown in the examples below.

Each of the scripts prompts the user to enter the password for the Sybase login.

4.8.7.3 Archive Utility Commands

The archive utility must be run with the following parameters in this exact order. There is only one command-line permutation:

```
DlDbArchiveAccessStat <MODE> <STARTDATE> <STOPDATE> <ARCHIVEDIR> <USERNAME> <SERVER> <DBNAME>
```

Example:

```
DlDbArchiveAccessStat OPS 20020405 20020505 /home/DBArchive/DataPool/ Labuser01
SybSQL_srvr DataPool_DB
```

The above example archives data to files and stores them in a specified directory.

4.8.7.4 Delete Utility Commands

The Delete Utility must be run with the following parameters in this exact order. There is only one command-line permutation:

```
DlDbDeleteAccessStat <MODE> <STARTDATE> <STOPDATE> <USERNAME> <SERVER> <DBNAME>
```

Example:

```
DlDbDeleteAccessStat OPS 20020912 20020913 Labuser01 SybSQL_srvr DataPool_DB
```

The above example deletes data in a specified time range.

4.8.7.5 Restore Utility Commands

The Restore utility must be run with the following parameters in this exact order. There is only one command-line permutation:

```
DlDbRestoreAccessStat <MODE> <STARTDATE> <STOPDATE> <ARCHIVEDIR> <USERNAME> <SERVER>
<DBNAME>
```

Example:

```
DlDbRestoreAccessStat OPS 20020405 20020505 /home/DBArchive/DataPool/ Labuser01
SybSQL_srvr DataPool_DB
```

The above example restores data in a specified time range from a specified archive directory.

4.8.7.6 Data Access Statistics Main Screen

The Data Pool Access Statistics utility does not have a main screen. It has a command line interface.

4.8.7.7 Required Operating Environment

The maintenance utilities run on a Linux platform.

4.8.7.7.1 Interfaces to Supporting Products

Table 4.8.7-2 lists the supporting products that these tools depend upon to function properly.

Table 4.8.7-2. Interface Protocols

Product Dependency	Protocols Used	Comments
DataPool database	SQL	Via SQL server machines

4.8.7.8 Databases

The Data Pool Access Statistics utility uses the DataPool database.

4.8.7.9 Special Constraints

The maintenance utilities run only if the Data Pool database is available and the Sybase server is running.

4.8.7.10 Outputs

There are no outputs from the maintenance scripts, except the error messages to the log.

4.8.7.11 Event and Error Messages

All error messages are written to the log files, which are DlDbRestoreAccessStat.log, DlDbArchiveAccessStat.log and DlDbDeleteAccessStat.log.

4.8.7.12 Reports

None.

4.8.8 Most Recent Data Pool Inserts Utility

The most recent data pool insert utility provides the ECS Operations Staff with a command-line interface for listing the most recent additions to data pool. Output of this utility is a set of files that a user could download and quickly inspect for new Data Pool additions. In operation, this utility would be configured to run as a cron job.

Utility takes in a date command-line parameter representing the day user is interested in. Files inserted into Data pool on this day would be listed in the output files. If a date is not provided, the utility uses the previous date as a default with a time range of midnight to midnight.

Since this utility requires connection to database, there will be a configuration file containing all information needed for accessing database. Further more, all error messages would be written to an error log file in /usr/ecs/<MODE>/CUSTOM/logs directory.

4.8.8.1 Using Most Recent Data Pool Inserts Utility

Utility would mainly be run as a cron job.

For command line usage, utility is started by entering the following:

```
> EcDIMostRecentInsert.pl <MODE> [-insertDate <YYYY/MM/DD>]
```

Command line parameters and corresponding descriptions that could be used with Most Recent Data Pool Inserts utility are listed in Table 4.8.8-1.

**Table 4.8.8-1. Command Line Parameter
Most Recent Data Pool Inserts Utility**

Parameter Name	Required	Description
MODE	Yes	An input parameter that specifies the mode of operation. This must be the first parameter passed, and it must be a valid, existing Data Pool mode such as OPS or TS1.
insertDate	No	An optional parameter specifying date in which user is interested. If date parameter is not present, previous day date is used by default. Date format is YYYY/MM/DD.

Executing this utility requires the mode as the first input parameter else a fatal error would be returned. If the date parameter is present, it must conform to the following format YYYY/MM/DD. Incorrect input parameters would result in errors being written to log file.

4.8.8.2 Most Recent Data Pool Inserts Utility Commands

Examples of how to use this utility is shown below:

1. EcDIMostRecentInsert.pl OPS -insertDate 2003/02/28

Queries database and creates file containing listings Data Pool additions for day 2003/02/28 for OPS mode.

2. EcDIMostRecentInsert.pl OPS

Since “-insertDate” command line parameter is not entered, the previous day is

used by default. Queries database and creates files listing recent additions to database for previous day for OPS mode.

4.8.8.3 Required Operating Environment

The O/S requirement is Linux 2.x.

4.8.8.4 Interfaces and Data Types

Table 4.8.8.2 lists the supporting products that this tool depends upon in order to function properly.

Table 4.8.8-2. Interface Protocols

Product Dependency	Protocols Used	Comments
Data Pool database	SQL	Via SQL server machines

4.8.8.5 Configuration File Format – EcDIMostRecentInsert.CFG

The “config” file contains vital details about how to connect to the Sybase database. Without this file, the utility can not run. The config file must be a single-entry plain text ASCII file, which has the following format:

```
SYB_USER = <string>
SYB_SQL_SERVER = <string>
PGM_ID = <string>
SYB_DBNAME = <string>
NUM_RETRIES = <integer>
SLEEP_SEC = <integer>
```

Breakdown of the individual parameters:

Parameter Name	Description
SYB_USER	The user name for the Sybase connection.
SYB_SQL_SERVER	The name of the SQL server for this Sybase connection.
SYB_DBNAME	The name of the database you intend to connect to
PGM_ID	Program ID used for connecting to the Data Pool database.
NUM_RETRIES	The number of times the utility will attempt to connect to the database before exiting. The recommended default is 5.
SLEEP_SEC	The number of seconds the utility will wait ('sleep') between connection attempts. The recommended default is 10.

4.8.8.6 Special Constraints

EcDIMostRecentInsert utility runs only if the Data Pool database server is running and if the database is available. It also assumes the stored procedures are present.

4.8.8.7 Outputs

Output of this utility is a set of files. One file located at top level Data Pool directory named DPRRecentInserts_<YYYYMMDD> and a file in each of the collection level directories named DPRRecentInserts_<ShortName>_<VersionID>_<YYYYMMDD>.

File DPRRecentInserts_<YYYYMMDD> contains distinct ShortNames and VersionIds while file DPRRecentInserts_<ShortName>_<VersionID>_<YYYYMMDD> contains ShortName, VersionId and fully qualified.

Note: The EcDIMostRecentInsert.pl would shut down and log an error message if it is unable to create a file at the top level data pool directory. If it is unable to create file at the collection level directory, program would log an error message and continue with processing other valid directories. Also, each time utility runs with the same input argument, the contents of the previously created file are over written.

4.8.8.8 Event and Error Messages

Usage and processing error messages are written to log file.

4.8.8.9 Reports

None.

4.8.8.10 Logs

The utility produces a log file called *EcDIMostRecentInsert.log* in the */usr/ecs/<MODE>/CUSTOM/logs* directory. If this log file already exists, the new information will automatically be appended. If there is no existing log file by this name, a new log file with this name will automatically be created.

4.8.8.11 Recovery

If there is an execution failure as a result of database server or system shut down, operator simply re-runs the script. This would create a new set of files (i.e. over writing previous ones) listing additions to Data Pool for the specified insert date.

4.8.8.12 Sybase Error Handling

The utility is highly dependent on Sybase server. Connection failure to Sybase Server would simply result in program termination and error logged to log file.

Note: The utility may repeatedly attempt to connect to the database, depending on how the configuration file was set. As an example, NUM_RETRIES set to 5 and SLEEP_SEC set to 10 in configuration file would mean the utility would try to connect 5 times, and will wait 10 seconds before each attempt.

This page intentionally left blank.

4.8.9 Data Pool Collection-to-Group Remapping Utility

The Data Pool Collection-to-Group Remapping Utility will allow DAAC Operations to re-assign a Data Pool collection to a collection group different from the one to which the collection was assigned originally. This command line utility will be used to remap collections between groups.

Note: Prior to using this utility you must set the “Insert Enabled Flag” to off using the Data Pool Maintenance GUI for the source collection.

4.8.9.1 Using the EcDIRemap utility

The Data Pool Collection-to-Group Remapping Utility is invoked as follows:

```
>EcDIRemap.pl <mode> -esdt <source collection name> -version <source collection version> -oldgrp <group to which the collection currently belongs> -newgrp <group to which the current collection will be mapped>
```

There are various command line parameters that are used in combination with each other. Table 4.8.9-1 provides a description of these parameters.

Table 4.8.9-1. Command Line Parameters

Parameter Name	Required	Description
mode	Yes	The mode in which the utility will run.
esdt	Yes	Specifies the name of the source collection that is being remapped
version	Yes	Specifies the version of the source collection that is being remapped
oldgrp	Yes	Specifies the name of the source collection group that contains the source collection
newgrp	Yes	Specifies the destination group where the source collection is to be mapped

Section 4.8.9.3 provides some examples along with detailed explanations for executing this utility.

4.8.9.2 Data Pool Collection-to-Group Remapping Utility Configuration File

The Data Pool Collection-to-Group Remapping utility uses a configuration file, EcDIRemap.CFG, located in /usr/ecs/<mode>/CUSTOM/cfg directory. The configuration parameters are stored in a PARAMETER = VALUE format with each parameter/value pair as a separate line entry in the file. Table 4.8.9-2 describes the configuration parameters.

Table 4.8.9-2. Configuration Parameters

Parameter Name	Value Description
SYB_USER	Sybase login name for the user of the Data Pool database.
SYB_SQL_SERVER	Name of Sybase SQL Server hosting Data Pool database.
SYB_DBNAME	Name of Data Pool database.
PGM_ID	Program identifier used as seed to generate database password.
NUM_RETRIES	Number of times database operation will be attempted.
SLEEP_SEC	Number of seconds between retries.

4.8.9.3 Examples for Remapping a Collection

1. **Remap a collection MOD29, Version 4 from the group MOST to the group MOSS in the OPS mode:**

```
EcDlRemap.pl OPS -esdt MOD29 -version 4 -oldgrp MOST -newgrp MOSS
```

The utility will remap the directory from the old collection MOD29.004 beneath the MOST group to the collection MOD29.004 under the MOSS group. The Data Pool database inventory will be updated to reflect the new location of the files.

Previous directory structure before remapping (example):

```
/datapool/OPS/user/MOST/MOD29.004/2000.10.31/MOD29.A2000305.h11v11.004.20012651  
13249.hdf
```

New directory structure following the remapping:

```
/datapool/OPS/user/MOSS/MOD29.004/2000.10.31/MOD29.A2000305.h11v11.004.20012651  
13249.hdf
```

4.8.9.4 Required Operating Environment

The Group remapping utility will run on a Linux platform.

4.8.9.5 Interfaces and Data Types

Table 4.8.9-3 lists the supporting products that this tool depends upon in order to function properly.

Table 4.8.9-3. Interface Protocols

Product Dependency	Protocols Used	Comments
Data Pool database	SQL	Via SQL server machines

4.8.9.6 Special Constraints

The Data Pool Collection-to-Group Remapping utility requires that the “Insert Enabled Flag” be turned off using the Data Pool Maintenance GUI for the source collection prior to running the

utility. If this is not done, the utility will provide an error message to the user indicating this condition and promptly exit. Also the utility expects that the group to which the user is mapping the collection exists in the Data Pool database. In addition, the user is to be aware that the utility does not allow remapping the Browse (Browse.001) collection. Data Pool functionality assumes that the browse collection is always located in the group "BRWS". Also, the utility doesn't allow users to map any collection to the BRWS group. The user is given an error message and the utility exits if either of these cases is true. The utility checks to see if the given collection is part of the old or source group. If not, the utility informs the user and exits. The Group Mapping utility runs only if the Data Pool database server is running and if the database is available. It also assumes the stored procedures are present.

The Remap Utility may not be used to remap QA or PH collections which have associated science collections. The science collections are associated with QA and PH collections through the Data Pool Maintenance Gui. If either the QA or PH collection is specified on the command line, the Remap utility will reject this input.

4.8.9.7 Outputs

Output of update events and errors will be always appended to a single log file.

4.8.9.8 Event and Error Messages

Usage errors will be displayed to the screen. Processing error messages are written to the log files.

4.8.9.9 Reports

None

4.8.9.10 Logs

The utility produces a log file called EcDIRemap.log in the /usr/ecs/<mode>/CUSTOM/logs directory. If this log file already exists, the new information will automatically be appended. If there is no existing log file by this name, a new log file with this name will automatically be created.

Since the log file may grow to a considerable size after constant use, it is recommended that it be saved off into a separate file from time to time for maintainability.

4.8.9.11 Recovery

The EcDIRemap.pl utility will be able to recover from aborted runs by using the DIRecoveryParameters table to checkpoint its progress. In the event of an aborted run, the utility will read the recovery parameters table to determine at which point the utility left off when it aborted. This will ensure that remappings that were taking place prior to the abort will finish correctly. After recovery processing takes place, the utility will then process the current run by acting on the latest input parameters. For example, if the original command line was:

```
EcDIRemap.pl OPS -esdt MOD29 -version 4 -oldgrp MOST -newgrp MOSS
```

If this run were aborted and a new invocation of the utility was initiated with a different collection and different source and destination groups as follows:

```
EcDIRemap.pl OPS -esdt AST_L1A -version 3 -oldgrp ASTT -newgrp ASTA
```

Then the utility would give a message to the standard output “Recovery detected. EcDirRemap.pl utility will now finish the previous run.” and log indicating a recovery was in process for remapping MOD29.004 collections from the old group (MOST) to the new group (MOSS) was taking place. As soon as the recovery was finished, the utility would indicate that it would now process the remapping of AST_L1A.003 from the ASTT group to the ASTA group.

4.8.9.12 Sybase Error Handling

If a Sybase error occurs, you will most likely see the actual Sybase error string displayed on the screen and in the log. Some errors can be that the database server is unavailable, that the connection to the database was dropped, or that there was an error executing the stored procedure. In the event of a Sybase-sourced error, the utility will immediately stop running.

In the event that a connection to the Data Pool database can not be established, the utility may repeatedly attempt to connect to the database, depending on how the configuration file was set. If, for example, NUM_RETRIES was set to 5 and SLEEP_SEC was set to 10, this means it will try to connect 5 times, and will wait 10 seconds before each attempt – a total of 50 seconds if all attempts are unsuccessful.

4.8.10 QA Update Utility

The QA Update Utility (QAUU) is an operational support tool used for updating the values of the QA flags. QA updates are applied to the XML metadata files in the metadata archive and DataPool file systems as well as to metadata tables in the DataPool database.

4.8.10.1 Using the QA Update Utility

The QAUU is started by executing the following:

```
$ EcDsAmQAUUStart <mode>  
  [ -file <filename> ] [ -noprompt ] [-noExitonError ]  
  [-recoverOnly | -abortRecovery | -skipRecovery | -recoverInvestigated]
```

All parameters, except for mode, are optional

<mode>: the mode to run the utility in

-file <filename>: run with a single request file where <filename> is the name of the request file, located in the configured request directory, containing the QA updates to be applied. Directory path is not allowed in the filename. If omitted, all request files in the configured request directory are processed.

-noprompt: if specified, the utility will not prompt the user for confirmations

-noexitonerr: if specified, the utility will not exit on the first error. This allows the operator to determine all errors that may occur during processing.

Recovery options:

These are all mutually exclusive; only one may be specified. Note that if none of these options are specified, the utility will recover, if necessary, and process new requests:

-recoverOnly: recover and do NOT process new requests (assume we do NOT recover failures flagged as “investigating”)

-abortRecovery: delete all failures (EXCEPT failures flagged as “investigating”) in working table and process new requests

-skipRecovery: flag (don't process) failures to be investigated (InvestigateFlag = 'Y') and process new requests

-recoverInvestigated: recover (including “investigating” failures) and process new requests

4.8.10.2 Required Operating Environment

Linux

4.8.10.3 Input File Format

FileName

The input, or “request”, file name has to follow the following naming convention, depending on the MODE and site where the request is from and when the request is generated:

```
<MODE>_<Site>_QAUPDATE<description>.<YY><MM><DD><hh><mm><ss>
```

where <description> is optional.

The following example shows the filename from site LDOPE for OPS mode at 12:20:30 on Feb. 29, 2008.

```
OPS_LDOPE_QAUPDATE_SetToPassed.20080229122030
```

Note: All the files in the request directory will be processed alphabetically by file name. This guarantees that all the requests coming from the same site will be processed in the right order.

When the DAAC operator copies files from the MODE-dependent FTP site to a MODE-dependent QAUURequest directory, he or she needs to copy all the files with filenames starting with that particular MODE. This way, we can minimize the risk of processing requests intended for a different MODE. QAUU also double checks the file name it processes in regard to the MODE.

In the situation when email is used to send a request, a file that follows the same naming convention should be created and attached to the email. Requests for different MODEs should be sent to different email aliases at each DAAC. The email script (EcDsQAUUEmailScript.pl) checks if the attachment file exists and if the file name follows the naming convention. If failed, the email script sends an email back to the requester indicating an error; otherwise it saves the email in the MODE-dependent QAUURequestDir directory using the same file name as the attachment file.

Note: We can not guarantee that email requests can arrive in sequence according to the time they are produced. Therefore, if the SCF needs to update the same granule(s) again for some reason, it needs to contact the DAAC operator to make sure that the previous requests for the same granule(s) have been finished.

Each request file must contain a **header**, a **body**, and a **footer**.

Request Header

For requests from FTP sites, the header contains one line indicating which site the request is from:

```
From <Site>
```

Example:

```
From LDOPE
```

For requests sent through emails, the headers are automatically generated, standard email headers.

Request Body

The body of the request contains a single **begin** statement, followed by the actual QA update **data** statement(s) that will be applied to the metadata.

The **begin** statement must be the first line in the body and is of this form:

```
begin QAMetadataUpdate {Science | Operational} {LGID | GranuleUR | ESDT}
```

The “Science” or “Operational” column tells the QAUU whether to update science or operational QA flags and explanations. The “LGID”, “GranuleUR” or “ESDT” column tells the QAUU whether the lines that follow the **begin** statement contain local granule ids (LGIDs), granule URs, or ESDT temporal ranges as criteria to look up the granules to be updated.

Each **data** statement takes the following form:

```
<ShortName>TAB<VersionID>TAB<granule search criteria>TAB<measured parameter name> |  
ALL TAB<QA flag value>TAB<QA flag explanation value>
```

Note that the columns for the **data** statement must be delimited by tabs and not spaces, since some of the columns can themselves contain spaces.

<granule search criteria> takes one of the following forms, depending on the LGID/GranuleUR/ESDT column in the **begin** statement.

For LGID, <granule search criteria> = the LGID of a specific granule to be updated.

For GranuleUR, <granule search criteria> = the GranuleUR of a specific granule to be updated.

For ESDT, <granule search criteria> = the temporal range of the granules to be updated.

<measured parameter name> can be an actual parameter name for the granule or the string “ALL”, which indicates that the QA flag change will be applied for all the granule’s parameters.

Examples:

(<**TAB**> indicates a required tab character in these examples.)

Request to update science QA flags based on LGID for specific parameters:

```
begin QAMetadataUpdate Science LGID  
GLA06<TAB>28<TAB>GLA06_428_2119_002_0221_4_01_0001.DAT<TAB>Surface Elevation<TAB>  
Passed<TAB>Comment about something  
GLA06<TAB>28<TAB>GLA06_420_2119_002_0221_4_02_0001.DAT<TAB>Surface Elevation<TAB>  
Failed<TAB>Another comment  
.....repeat for each LGID
```

Request to update operational QA flags based on GranuleUR for a specific parameter:

```
begin QAMetadataUpdate Operational GranuleUR  
AE_Land<TAB>86<TAB>UR:10:DsShESDTUR:UR:15:DsShSciServerUR:10[:DSSDSRV]:17:SC:AE_Land.086:74735<TAB>Surf  
ace Soil Moisture<TAB>Passed<TAB>Comment  
.....repeat for each GranuleUR
```

Request to update science QA flags based on ESDT and temporal range for all parameters:

```

begin QAMetadataUpdate Science ESDT
GLA06<TAB>28<TAB>Mar 31 2007<TAB>Apr 1 2007<TAB>ALL<TAB>Being Investigated<TAB>Comment
AE_Land<TAB>86<TAB>Mar 31 2007<TAB>Apr 1 2007<TAB>ALL<TAB>Failed<TAB>Comment on failure
.....repeat for each ESDT

```

Request Footer

The footer is simply a statement that indicates the end of the request file and must take the following form:

```
end QAMetadataUpdate
```

4.8.10.3.1 Configuration File Format

Table 4.8.10-1 shows the configuration file parameters.

Table 4.8.10-1. Configuration File Parameters (1 of 3)

Parameter Name	Description
SYBASE_SQL_HOST	The host for the Inventory and DataPool databases
SYBASE_SQL_SERVER	The name of the Sybase server for the Inventory and DataPool databases
SYBASE_JDBC_DRIVER_CLASS	The Java class used for connecting the QAUU Java application to Sybase
SYB_DBNAME	The name of the Inventory database
SYB_DPL_DBNAME	The name of the DataPool database
SYB_PORT	The port number used to connect to the Inventory database
SYB_USER	The username used to connect to and perform queries for the Inventory and DataPool databases
PGM_ID	The ECS Program ID for the QAUU user (SYB_USER)
DB_NUM_RETRIES	The number of times to retry failed DB operations
DB_SLEEP_SEC	The number of seconds between DB operation retries
EMAIL_SERVER_HOST	Host name where email server runs
EMAIL_SMTP_USER	Email SMTP user name
EMAIL_QAUU_FROM_ADDRESS	Email notification sender address
FILE_NUM_RETRIES	The number of times to retry failed file operations
FILE_SLEEP_SEC	The number of seconds between file operation retries
QA_REQUEST_DIR	Path of directory containing QA update requests

Table 4.8.10-1. Configuration File Parameters (2 of 3)

Parameter Name	Description
QA_ERROR_REQUEST_DIR	Path of directory containing QA update requests that have failed.
QA_COMPLETED_REQUEST_DIR	Path of directory containing successfully completed QA update requests
QA_TEMP_DIR	Path of directory containing temporary files
QA_HISTORY_DIR	Path of directory containing QA update history files
DAAC_EMAIL_ADDRESSES	List of valid DAAC email notification addresses
<SCFSite>_EMAIL_FROM_ADDRESSES	List of valid email notification from addresses for a <SCFSite>
<SCFSite>_EMAIL_REPLY_ADDRESS	Valid email notification reply address for a <SCFSite>
<SCFSite>_NOTIFICATION_ON_SUCCESS	Flag indicating (if = "Y") that email notification should be sent upon successfully processing QA update requests for a <SCFSite> or for requests that fail. If = 'N', email should only be sent for requests that fail.
VALID_SCIENCE_QA_FLAGS	List of valid science QA flag values
VALID_OPERATIONAL_QA_FLAGS	List of valid operational QA flag values
NUM_XML_THREADS	The number of threads to be used. One thread will operate upon an UPDATE_BATCH_SIZE of QA updates.
MAX_NUM_GRANULES	The maximum number of granules that can be updated per run
UPDATE_BATCH_SIZE	The number of QA updates to be performed at a time.
XML_ARCHIVE_DIRECTORY	Pathname of XML Archive file system
SOCKS_PROXY_HOST	SOCKS proxy hostname
SOCKS_PROXY_PORT	SOCKS proxy port
BCP_EXEC_PATH	Path to Unix bcp executable
SHELL_PATH	Path to Unix sh shell needed to perform Unix commands
application.name	Name of this application
log.operations.level	Level of logging desired in operational log: NONE, INFORMATION, VERBOSE or XVERBOSE
log.debug.level	Level of logging desired in debug log: NONE, INFORMATION, VERBOSE or XVERBOSE
log.performance.level	Level of logging desired in performance log: NONE, INFORMATION, VERBOSE or XVERBOSE
log.overwrite	If true, log file will be overwritten for each run
log.threshold	Size of log files before new ones are created.
log.rotation.number	Number of log files that will be rotated through.

Table 4.8.10-1. Configuration File Parameters (3 of 3)

Parameter Name	Description
XML_TEMP_DIR	QAUU working directory used to save the original xml file in case of failure. This should be a unmanaged directory within the XML archive for the purpose of reducing StorNext activities.

4.8.10.3.2 Recovery

The QA Update Utility offers the following mechanism for recovering from failures in a previous run. A working table is used to store the rows of QA updates to be worked off for a run. Rows are deleted when their updates are successfully applied. At the end of an uninterrupted run, all successfully updated rows will have been deleted from the working table. Rows for any updates which failed or did not complete will remain in the working table, and the reason for failure or incompleteness will be stored in a column of the table. After an error-free run, this working table will be empty. During startup, if the working table contains one or more rows, recovery is attempted. The following failure/incompletion states are recorded in the working table DsQAMUTRequestDetail statusFlag column, and they happen in the following sequence:

FAILED_XML_UPDATE(“X”): could not update the XML file in the XML Archive

UPDATED_XML(“U”): successfully updated the XML file in the XML Archive

FAILED_BMGT_UPDATE(“B”): could not update the information needed by BMGT.

UPDATED_BMGT(“M”): successfully updated the information needed by BMGT

FAILED_XML_COPY_TO_DPL(“C”): could not copy the XML Archive file to the Datapool

COPIED_XML(“P”): successfully copied the XML Archive file to the datapool

FAILED_DPL_DB_UPDATE: could not update the Datapool MeasuredParameter table

These states allow the utility to avoid duplicate work during recovery because they indicate what processing remains to be done. FAILED_XML_UPDATE indicates that all processing remains to be done. UPDATED_XML and FAILED_BMGT_UPDATE indicate that the XML updates have already been applied, but the updates for BMGT failed and needs to be retried. COPIED_XML indicates that the update of the DataPool MeasuredParameter table is all that remains to be done. If a failure is attempted and fails a second time, the InvestigateFlag in the working table DsQAMUTRequestDetail table will be set to “Y”. These are handled via various command-line options (see Section 4.8.10.1 Using the QA Update Utility above).

4.8.10.3.3 QA Update Email Script

There are two ways to place QA update request files in the QAUU request directory (the configured QA_REQUEST_DIR directory): manually and via email. Remote sites can submit update request files via email as attachments. With proper configuration, the email server will detect the emails as QAUU requests and invoke a Perl script - EcDsQAUUEmailScript.pl. The

script parses the request, grabs the attached request file and moves it to the QAUU request directory. It resides on the central mail servers while the QAUU resides on other boxes. The directories containing the email script output (/usr/ecs/<mode>/CUSTOM/data/DSS/QAUU and its subdirectories) will be created on the hosts running the QAUU and remote mounted on the central mail servers. Special QAUU email aliases need to be set up in the /etc/aliases file on the email server host to direct email QAUU update request to the email script. One email alias is required for each mode supporting QAUU:

```
QAUU_<MODE>: "| /usr/ecs/<MODE>/CUSTOM/utilities/EcDsQAUUEmailScript.pl"
```

4.8.10.3.4 Outputs

Output of update events and errors will be always appended to a single log file. If specified as an option, a confirmation prompt will be displayed to the screen.

4.8.10.3.5 Event and Error Messages

If prompting is specified on the command line (default), confirmation messages are displayed to the operator. Otherwise, all prompts and screen displays are suppressed. All the error messages are written in the log file except for command line syntax errors which are displayed on the screen.

4.8.10.3.6 Reports

None

This page intentionally left blank.

4.8.11 Data Pool Move Collections Utility

The Move Collections Utility provides the EMD Operations Staff with a command-line interface to move Data Pool collections from one file system to another. The utility requires command-line parameters that specify the collection (shortname and version id) to be moved and the source and target file system labels. The utility also supports `-verbose` and `-debug` command line options which control the amount of information logged by the utility. The `-verbose` option allows detailed information, such as copy and remove commands, to be written to the utility log. The `-debug` option causes the utility to print additional information about utility parameters to the log file. The default for these options are non-verbose and non-debug.

Fault recovery is also supported, allowing completion of a collection move which was interrupted due to a database server fault or an operating system error.

A collection move is implemented via the following series of operations:

1. The utility sets the `moveFlag` for the collection in the `DICollections` table, to indicate that a collection move is in progress. While the `moveFlag` is set, Data Pool publishing for the collection is suspended (although granules in the collection may still be registered in the Data Pool, e.g., during ingest). In addition, while the `moveFlag` is set, access to the collection via the Data Pool Web Access GUI will be prevented. The Web Access GUI will not display the collection name as an active link on the collection drill down web page, and access to the collection via previously stored user bookmarks will also be prevented.
2. The utility performs a copy operation of all files in the collection to the new collection directory structure on the target file system. During the copy operation, all files and links in the original public directory structure for the collection are copied to the new public directory structure for the collection (this includes science files, xml files, browse links, and Most Recent Inserts files). All files and links in the original hidden directory structure for the collection are copied to the new hidden directory structure for the collection.
3. When the copy operation completes successfully, the `fileSystemLabel` in the `DICollections` table is updated with the target file system label. Once the `DICollections` table is updated, subsequent Data Pool Ingest and Insert operations will write to the target file system, and subsequent distribution operations will read from the target file system. In addition, the update to `DICollections` will trigger a BMGT export of the updated Data Pool URLs for all files in the collection. Note that all URL updates for the collection will be contained in a single export package.
4. The utility then proceeds to remove the old collection directory structure and its contents. This operation is performed in conjunction with a rescan of the directory structure for “straggler” granules which were inserted after the Move Collections utility began its copy operation. If such files are found, they are copied to the collection directory structure in the target file system before being removed from the source collection directory structure. This check for stragglers may be executed additional times, depending on the values of the `NUM_DELETE_RETRIES` and `DELETE_SLEEP_SEC` parameters in the Move Collections utility configuration file.
5. When the old collection directory structure and its contents have been completely removed, the utility sets a symbolic link from the old collection-level directory to the new one. For

example, before a move, a collection might be located in the /datapool/OPS/user/FS1/MOAT/AIRABRAD.007 directory structure. After invoking the utility with a target file system of FS2, all files associated with the collection will be moved to the /datapool/OPS/user/FS2/MOAT/AIRABRAD.007 directory structure. A symbolic link will be created in the old collection-level directory, pointing to the new collection-level directory, i.e. /datapool/OPS/user/FS1/MOAT/AIRABRAD.007
→/datapool/OPS/user/FS2/MOAT/AIRABRAD.007

These links will be persistent so that the data in the collection can still be retrieved using the original URL. DAACs can remove the link via unix commands when they believe that it is no longer needed. Otherwise, the link will only be removed (by the Data Pool Cleanup Utility) if the collection (in its new location) becomes empty.

6. Reset the moveFlag to “N” in the DICollections table, to indicate that the move has completed.

Note that existing URLs and file pointers will be invalid from the time when the utility removes the associated files from the source directory structure until the time the collection-level symbolic link is established. During this time:

- A Data Pool FTP user clicking on a URL might experience a temporary error when trying to access files and directories associated with the moving collection. File transfers that are already in progress when deletion begins should complete normally.

NOTE: The DAAC may wish to alert FTP users of the unavailability of a moving collection via a broadcast message on the DAAC or Data Pool home page, and may wish to take additional measures to prevent FTP access to the moving collection, such as changing the directory permissions.

- A WIST (or other ECHO client) user clicking on a Data Pool URL might experience a temporary error when trying to access files and directories associated with the moving collection. File transfers that are already in progress when deletion begins should complete normally.
- FTP Pull users could experience similar temporary problems when they try to access links in FTP Pull directories that were established by the OMS and that point to granules in the moving collection.

In addition, the following errors may occur during a collection move:

- The Order Manager Server looks up granule file locations immediately before performing an FTP Push operation. If the lookup occurs just before the collection information in the Data Pool database is updated, but the copy operation starts after the file was deleted, the FTP Push operation will fail and cause an operator intervention. Since the interval of time between file location look up and ftp push start is small, the chances that this error will occur are very small. If the error does occur, the operator should resubmit the distribution request, and since the directory entry will now have been updated, the ftp push operation will succeed.

- Data Pool Ingest accesses granule files several times during an ingest operation, and granules in the collection that are in active ingest requests at the time when the fileSystemLabel is updated in DICollections may experience an error because Data Pool Ingest cannot locate them. This would result in an operator intervention. **Restarting** the granule from the operator intervention after the collection move completes successfully will resolve the error.

4.8.11.1 Using the Move Collections Utility

The Move Collections Utility is started by entering the following command:

```
> EcDIMoveCollection.pl <mode> -shortname <shortname>
                        -versionid <versionid> -sourcefs <source file system label>
                        -targetfs <target file system label> -verbose -debug
```

There are various command line parameters that are used in combination with each other. Table 4.8.11-1 provides a description of these parameters.

Table 4.8.11-1. Command Line Parameters of the Move Collections Utility (1 of 2)

Parameter Name	Description
<mode>	An input parameter that specifies the mode of operation. This must be the first parameter passed, and it must be a valid, existing Data Pool mode with a format like OPS or TS1. This parameter is mandatory. Note: The user will be prompted if the utility is run in OPS mode to prevent any accidental loss of data.
-verbose	Directs the utility to run using verbose option. Some information will be displayed to the screen and detailed information will be written to the utility's log. Default is nonverbose.
-shortname <shortname>	An input parameter that specifies the shortname of the collection to be moved. This parameter is mandatory. Note that the utility may not be used to move the Browse collection.
-versionid <versionid>	An input parameter that specifies the version identifier of the collection to be moved. Do not specify leading zeros. This parameter is mandatory.
-sourcefs <file system label>	An input parameter that specifies the label of the source file system (e.g., FS1, WORKING, LONGTRM) from which the collection is being moved. Note that all Data Pool file systems must be mounted under the Data Pool root (e.g. (/datapool/OPS/user)). This parameter is mandatory.
-targetfs <file system label >	An input parameter that specifies the label of the target file system path to which the collection is being moved (e.g., FS1, WORKING, LONGTRM). Note that all Data Pool file systems must be mounted under the Data Pool root (e.g. (/datapool/OPS/user)). This parameter is mandatory.

Table 4.8.11-1. Command Line Parameters of the Move Collections Utility (2 of 2)

Parameter Name	Description
-debug	This option directs the utility to include additional debug information in the log. Default is no debug.

There is no required ordered sequence of the command line parameters except for the parameter <MODE>. This must be first parameter or a fatal error will be returned. The combination of the remaining command line parameters must be valid. A command line input error results in a 'usage' syntax display and in most cases will also explain why the command line input was incorrect.

4.8.11.2 Move Collections Utility Commands

Below are some examples for invoking this utility:

```
1. EcDlMoveCollection.pl <mode> -shortname MODVOLC  
-versionid 1 -sourcefs FS1 -targetfs FS2 -verbose
```

Moves the files, browse links, and inventory information for the collection **MODVOLC.001** from its current directory structure in file system FS1 to the new filesystem FS2. The collection will be moved from /datapool/<mode>/user/FS1/MOAT to /datapool/<mode>/user/FS2/MOAT. Files in the hidden directory structure /datapool/<mode>/user/FS1/.orderdata/<encrypted MOAT> will be moved to /datapool/<mode>/user/FS2/.orderdata/<encrypted MOAT>. The utility will be run using the -verbose option, which displays information to the screen and to the log.

```
2. EcDlMoveCollection.pl <mode> -shortname MODVOLC  
-versionid 1 -sourcefs FS1 -targetfs FS2
```

Same as 1) but in non-verbose mode. No output to the screen and less detail in the log.

4.8.11.3 Required Operating Environment

The Move Collections Utility will run on a Linux platform from which Sybase and the Stornext Data Pool file systems are accessible.

4.8.11.4 Interfaces and Data Types

Table 4.8.11-2 lists the supporting products that this tool depends upon in order to function properly.

Table 4.8.11-2. Interface Protocols

Product Dependency	Protocols Used	Comments
Data Pool database	SQL	Via SQL server machines
Perl DBI	DBD::Sybase	Requires proper install of baselined version of Perl.

4.8.11.5 Input File Format

N/A

4.8.11.6 Configuration File Format – EcDIMoveCollection.CFG

The “config” file contains vital details about how to connect to the Sybase database. Without this file, the utility can not run. The config file must be a single-entry plain text ASCII file, which has the following format:

```
SYB_USER = <string>
SYB_SQL_SERVER = <string>
SYB_DBNAME = <string>
PGM_ID = <string>
NUM_DB_RETRIES=<integer>
DB_SLEEP_SEC=<integer>
NUM_DELETE_RETRIES=<integer>
DELETE_SLEEP_SEC=<integer>
```

See Table 4.8.11-3 for a breakdown of individual parameters.

Table 4.8.11-3. Configuration File Parameters

Parameter Name	Description
SYB_USER	The user name for the Sybase connection.
SYB_SQL_SERVER	The name of the SQL server for this Sybase connection.
SYB_DBNAME	The name of the database you intend to connect to
PGM_ID	Program ID used for connecting to the Data Pool database. The value of this parameter must be set to 10000022 for this program.
NUM_DB_RETRIES	The number of times the utility will attempt to connect to the database before exiting. The recommended default is 5.
DB_SLEEP_SEC	The number of seconds the utility will wait ('sleep') between connection attempts. The recommended default is 10.
NUM_DELETE_RETRIES	The number of times the utility will rescan the old collection directory prior to deleting it. If the delete fails, it is most likely because the directory is not empty because some granules were inserted after the move started. The repeated rescanning for these files handles this case. The recommended default is 5.
DELETE_SLEEP_SEC	The number of seconds the utility will wait ('sleep') between old collection directory rescans/deletes. The recommended default is 10.

4.8.11.7 Special Constraints

- The Move Collections Utility runs only if the Data Pool database server is running and if the database is available. It also assumes the stored procedures are present.
- The Move Collections Utility and the QA Update Utility are not allowed to run simultaneously. Each utility queries the database to see if the other is running and will shut down if the other is running.

- The Move Collections Utility should not be run at the same time as the Data Pool Cleanup Utility, if the Data Pool Cleanup utility run will clean up granules in the collection being moved.
- Data Pool publications for granules in the moving collection are suspended while the move is in progress, and resumed when the move completes.
- The Move Collections Utility may not be used to move the Browse collection. If the Browse collection is specified on the command line, the utility will reject this input.
- The Move Collections Utility may not be used to move QA or PH collections which have associated science collections. The science collections are associated with QA and Ph through the Data Pool Maintenance Gui. If either the QA or PH collection is specified on the command line, the utility will reject this input.

Note: Only one instance of the Move Collections Utility should be run at a time. Running multiple instances concurrently, whether on the same or different hosts, may result in system resource conflicts and/or fault recovery conflicts.

4.8.11.8 Outputs

Output of update events and errors for a collection will be always appended to a single log file named by data type. See section 4.8.11.11.

4.8.11.9 Event and Error Messages

Usage errors will be displayed to the screen. Processing error messages are written to the log files.

4.8.11.10 Reports

None

4.8.11.11 Logs

The utility produces a log file called EcDIMoveCollection<ShortNameVersionId>.log in the /usr/ecs/<mode>/CUSTOM/logs directory. If this log file already exists, the new information will automatically be appended. If there is no existing log file by this name, a new log file with this name will automatically be created.

Since the log file may grow to a considerable size after constant use, it is recommended that it be saved off into a separate file from time to time for maintainability.

4.8.11.12 Recovery

The Move Collections Utility provides a capability to recover from an execution failure caused by situations such as system faults or database errors leaving all or some of the file moves unprocessed. At startup, the utility will determine whether a previous run has failed to complete. If so, the operator will be prompted as to whether or not to attempt recovery of the incomplete run. If the failure in the incomplete run occurred BEFORE the fileSystemLabel for the collection was set to the target file system, the operator will have three options: 1) continue the move of the interrupted collection; 2) defer this recovery; or 3) do not recover the interrupted collection. If the failure occurred AFTER the fileSystemLabel for the interrupted collection was

set to the target file system, the operator will have only two options: 1) continue the move of the interrupted collection; or 2) defer this recovery.

If the operator chooses to recover, the utility will complete the move operations for the collection in the previous incomplete run.

If the operator chooses to defer recovery, the utility will exit, and the moveFlag for the interrupted collection and the recovery parameters for the incomplete run will not be changed. The operator will not be able to proceed with moves of additional collections until the interrupted collection is eventually completed.

If the operator chooses not to recover the interrupted collection move, the utility will reset the moveFlag for the interrupted collection to “N”, clear the recovery parameters for the incomplete run, and, if a new collection was specified on the command line, the utility will proceed with the move of that collection. It will be up to the operator to manually clean up any files from the incomplete collection move that have been moved to the target file system.

4.8.11.13 Sybase Error Handling

If a Sybase error occurs, the operator will most likely see the actual Sybase error string displayed on the screen and in the log. Possible Sybase errors include that the database server is unavailable, that the connection to the database was dropped, or that there was an error executing a stored procedure. In the event of a Sybase-sourced error, the utility will immediately stop running.

In the event that a connection to the Data Pool database can not be established, the utility may repeatedly attempt to connect to the database, depending on how the NUM_RETRIES and DB_SLEEP_SEC parameters in the utility configuration file were set. If, for example, NUM_RETRIES was set to 5 and DB_SLEEP_SEC was set to 10, the utility will try to connect to the database 5 times, and will wait 10 seconds before each attempt – a total of 50 seconds if all attempts are unsuccessful.

This page intentionally left blank.

4.8.12 Data Pool Hidden Scrambler Utility

The Data Pool Hidden Scrambler utility provides a mechanism by which the ECS Operations Staff can encrypt or re-encrypt the names of Data Pool hidden directories, both on the file system and in the Data Pool database.

The Data Pool Hidden Scrambler utility may be run with either the “transition” option (one time only, when hidden directory names are first created in the database for all Data Pool collections), or the “rename” option (when hidden directory names need to be re-encrypted, either to respond to a security breach, or on a scheduled basis at the DAAC, depending on DAAC security policy). This utility should be run as `cmshared`, `cm<mode>`, or similar.

4.8.12.1 Using the Data Pool Hidden Scrambler Utility

The Data Pool Hidden Scrambler utility should be started by the user `cmshared` (or similar). The Data Pool Hidden Scrambler utility is started by entering the following command:

```
EcDIHiddenScramblerDataPool.pl <mode> <command line parameters>
```

There are four command line parameters that may be used. Table 4.8.12-1 provides a description of those parameters.

Table 4.8.12-1. Command Line Parameter

Parameter Name	Required	Description
transition	No	This parameter may not be used with any of the other command line parameters. Specifies that the utility should be run with the transition option.
collgroup	No	This parameter may not be used with the “transition” parameter, nor with the “shortname”/“versionid” parameters. Specifies that the utility should be run with the rename option, for all collections in the indicated collection group.
shortname	No	This parameter may not be used with the “transition” parameter, nor with the “collgroup” parameter. It must be used with the “versionid” parameter. Specifies that the utility should be run with the rename option for the indicated collection only.
versionid	No	This parameter may not be used with the “transition” parameter, nor with the “collgroup” parameter. It must be used with the “shortname” parameter. Specifies that the utility should be run with the rename option for the indicated collection only.

The Hidden Scrambler utility performs the following as part of the “rename” processing:

- Generates a new random `orderOnlySNDirName` and `orderOnlyGrpDirName` for each requested Collection and Collection Group in the Data Pool, and saves these names to the Data Pool database.

Note: If the `collgroup` parameter is used then the utility generates a new random `orderOnlyGrpDirName` for the collection group supplied and generates a new random

orderOnlySNDirName for each collection in that collection group. If the shortname and versionid parameters are used, then the utility will only generate a new random orderOnlySNDirName for the specified collection, and save that to the database. If neither the collgroup nor shortname/versionid parameters are used, the utility generates a new random orderOnlyGrpDirName for all collection groups in the mode and generates a new random orderOnlySNDirName for all collections in the mode.

- Creates new hidden directories based on the new orderOnlySNDirName and orderOnlyGrpName for each requested collection.
- Copies all files from the old hidden directories to the newly created hidden directories.
- Updates the FTPpull links for existing orders referencing the old hidden directories, to point to the new hidden directories
- Removes the old hidden directories.
- Reports the time it takes to update the FTPpull links.

The Data Pool Hidden Scrambler utility performs the following as part of the "transition" processing:

- Generates a new random orderOnlySNDirName and orderOnlyGrpDirName for each Collection and Collection Group in the Data Pool, and saves these names to the Data Pool database

4.8.12.1.1 Hidden Scrambler Utility Command Line Examples

1. For a "rename" run:

Note: For a "rename" run, the Hidden Scrambler utility should only be run during Data Pool downtime. The script must be run with a user account with privileges to rename directories on the Data Pool.

```
EcDlHiddenScramblerDataPool.pl OPS
```

The Hidden Scrambler Utility will perform rename processing for all collection groups and all collections in the Data Pool in OPS mode.

```
EcDlHiddenScramblerDataPool.pl OPS -collgroup MOAT
```

The Hidden Scrambler Utility will perform rename processing for the MOAT collection group and for all collections in the MOAT collection group, in OPS mode.

```
EcDlHiddenScramblerDataPool.pl OPS -shortname AST_L1B -versionid 3
```

The Hidden Scrambler Utility will perform rename processing only for the AST_L1B.003 collection in OPS mode. (Note that the corresponding collection group (ASTT) hidden directory name will not be re-encrypted).

2. For a "transition" run:

Note: Transition may be used while Data Pool is up. It should be used only once, the first time the utility is run in any given mode.

EcDIHiddenScramblerDataPool.pl TS1 -transition

The Hidden Scrambler Utility will generate encrypted directory names for all Data Pool collections and collection groups in TS1 mode, and save the names in the Data Pool database.

4.8.12.2 Hidden Scrambler Configuration File

The Data Pool Hidden Scrambler utility uses a configuration file, EcDIHiddenScrambler.CFG, located in /usr/ecs/<mode>/CUSTOM/cfg directory. The configuration parameters are stored in a PARAMETER = VALUE format with each parameter/value pair as a separate line entry in the file. Table 4.8.12-2 describes the configuration parameters.

Table 4.8.12-2. Configuration Parameters

Parameter Name	Value Description
SYB_USER	Sybase login name for the user of the Data Pool database.
SYB_SQL_SERVER	Name of Sybase SQL Server hosting Data Pool database.
SYB_DBNAME	Name of Data Pool database.
PGM_ID	Program identifier used as seed to generate database password.
NUM_RETRIES	Number of times database operation will be attempted.
SLEEP_SEC	Number of seconds between retries.
PULL_DIR	Location of the FTP Pull Directory in appropriate mode. NOTE: Be sure to use the full path to the FTP Pull Directory, not a linked path (e.g. /datapool/<mode>/user/<fs>/PullDir).

4.8.12.3 Data Pool Hidden Scrambler Utility Main Screen

The Data Pool Hidden Scrambler Utility does not have a main screen. It has a command line interface only.

4.8.12.4 Required Operating Environment

The Hidden Scrambler Utility will run on a Linux platform.

4.8.12.5 Databases

Table 4.8.12-3 lists the supporting products that this tool depends upon in order to function properly.

Table 4.8.12-3. Product Dependencies

Product Dependency	Protocols Used	Comments
Data Pool database	SQL	Via SQL server machines

4.8.12.6 Special Constraints

The Data Pool Hidden Scrambler utility runs only if the Data Pool database server is running and if the database is available. It also assumes the stored procedures are present.

With the rename option, the utility must be run during Data Pool downtime.

The utility should only be run once with the transition option, the first time the utility is run in any given mode.

4.8.12.7 Outputs

Output of update events and errors will be always appended to a single log file.

4.8.12.8 Event and Error Messages

Usage errors will be displayed to the terminal screen. Processing error messages are written to the log files.

4.8.12.8 Reports

None.

4.8.12.9 Logs

The utility produces a log file called EcDIHiddenScrambler.log in the /usr/ecs/<mode>/CUSTOM/logs directory. If this log file already exists, the new information will automatically be appended. If there is no existing log file by this name, a new log file with this name will automatically be created.

Since the log file may grow to a considerable size after constant use, it is recommended that it be saved off into a separate file from time to time for maintainability.

4.8.12.11 Recovery

The Data Pool Hidden Scrambler Utility provides a capability to recover from interruptions caused by situations such as the system faults or database errors leaving all or some of the directories unprocessed. The utility will detect such failure upon the next run and continue processing the directories and files that were left unprocessed in the previous run. The operator is given no choice as to recovery. Recovery will proceed so that the Data Pool inventory and disk files will not be left in a corrupted state.

4.8.12.12 Sybase Error Handling

If a Sybase error occurs, the actual Sybase error string will most likely be displayed on the screen and in the log. Possible errors include that the database server is unavailable, that the connection to the database was dropped, or that there was an error executing a stored procedure. In the event of a Sybase-sourced error, the utility will immediately stop running.

In the event that a connection to the Data Pool database cannot be established, the utility may repeatedly attempt to connect to the database, depending on how the configuration file was set. If, for example, NUM_RETRIES was set to 3 and SLEEP_SEC was set to 10, the utility will try to connect to the database 3 times, and will wait 10 seconds between each attempt – a total of 30 seconds if all attempts are unsuccessful.

4.8.13 Data Pool Remove Collection Utility

The Data Pool Remove Collection utility provides a mechanism by which ECS Operations staff can remove collections from the Data Pool database that are no longer of interest to the end users.

4.8.13.1 Using the Data Pool Remove Collection Utility

The Data Pool Remove Collection utility is started using the following parameters:

```
EcDIRemoveCollection.pl <MODE> -ShortName <SHORTNAME> -VersionId <VERSIONID> [-debug]
```

OR

```
EcDIRemoveCollection.pl <MODE> -inpfile <INPUTFILENAME> [-debug]
```

Table 4.8.13-1 lists the descriptions of the command line parameters.

Table 4.8.13-1. Command Line Parameters

Parameter Name	Required	Description
debug	No	Helps developers debug the app by printing copious debug information
ShortName	Yes	ShortName of the collection to be deleted
VersionId	Yes	VersionId of the collection being deleted
inpfile	Yes	The full path to an input file specifying multiple collections. Please note that either an input file or a ShortName/VersionId combination should be used but not both

The input file contains a list of ShortName VersionId pairs, one pair per line, as shown below:

```
ShortName1 VersionId1
```

```
ShortName2 VersionId2
```

...

There should be at least one space or tab between the ShortName and VersionId on each line in the input file. Other white space will not affect the utility.

If there are any active granules associated with the collection, or if there are any other database errors, the utility will print an appropriate error message on the screen and log the message too.

The Remove Collection utility removes collections only from the Data Pool database. The ECS Operations staff is responsible for removing any directories (public and hidden) associated with the collection from the Data Pool file system.

4.8.13.2 Remove Collection Configuration File

The Data Pool Remove Collection utility uses a configuration file, EcDIRemoveCollection.CFG, located in the /usr/ecs/<mode>/CUSTOM/cfg directory. The configuration parameters are stored in a PARAMETER = VALUE format with each parameter/value pair as a separate line entry in the file. Table 4.8.13-2 describes the configuration parameters.

Table 4.8.13-2. Configuration Parameters

Parameter Name	Value Description
SYB_USER	Sybase login name for the user of the Data Pool database.
SYB_SQL_SERVER	Name of Sybase SQL Server hosting Data Pool database.
SYB_DBNAME	Name of Data Pool database.
PGM_ID	Program identifier used as seed to generate database password.
NUM_RETRIES	Number of times database operation will be attempted.
SLEEP_SEC	Number of seconds between retries.

4.8.13.3 Data Pool Remove Collection Utility Main Screen

The Data Pool Remove Collection Utility does not have a main screen. It has a command line interface only.

4.8.13.4 Required Operating Environment

The Data Pool Remove Collection Utility will run on a Linux platform. It assumes that perl with the Sybase DBI modules is already installed.

4.8.13.5 Databases

Table 4.8.13-3 lists the supporting products that this tool depends upon in order to function properly.

Table 4.8.13-3. Product Dependencies

Product Dependency	Protocols Used	Comments
Data Pool database	SQL	Via SQL server machines

4.8.13.6 Special Constraints

The Data Pool Remove Collection utility runs only if the Data Pool database server is running and if the database is available. It also assumes the stored procedures are present.

4.8.13.7 Outputs

Output of collection removal events and errors will be always appended to a single log file. See Section 4.8.16.8.

4.8.13.8 Event and Error Messages

Usage errors will be displayed to the screen. Processing error messages are written to the log files.

4.8.13.9 Reports

None

4.8.13.10 Logs

The utility produces a log file called EcDIRemoveCollection.log in the /usr/ecs/<mode>/CUSTOM/logs directory. If this log file already exists, the new information will automatically be appended. If there is no existing log file by this name, a new log file with this name will automatically be created.

Since the log file may grow to a considerable size after constant use, it is recommended that it be saved off into a separate file from time to time for maintainability.

4.8.13.11 Recovery

Since the removal of a collection is handled in one database transaction, it either works or it does not. Hence there is no need for recovery.

4.8.13.12 Sybase Error Handling

If a Sybase error occurs, the operator will most likely see the actual Sybase error string displayed on the screen and in the log. Some errors can be that the database server is unavailable, that the connection to the database was dropped, or that there was an error executing the stored procedure. In the event of a Sybase-sourced error, the utility will immediately stop running.

In the event that a connection to the Data Pool database can not be established, the utility may repeatedly attempt to connect to the database, depending on how the configuration file was set. If, for example, NUM_RETRIES was set to 5 and SLEEP_SEC was set to 10, this means it will try to connect 5 times, and will wait 10 seconds before each attempt – a total of 50 seconds if all attempts are unsuccessful.

This page intentionally left blank.

4.8.14 Data Pool Band Backfill Utility

The DPL Backfill Utility is a command line tool that can correct band extraction problems that occurred during DPL registrations. Granule registrations cannot fail if band extraction problems are encountered but the subsequent publications on convert-enabled data types must fail if the band information is not present in the DPL database at publication time.

The Band Backfill utility was developed to correct the problems above. It will:

- backfill the band information in the DPL database for the registered granules specified in its input file.
- request the publication of the backfilled granules via the new Data Pool Action driver.

The DAAC Operations staff can identify the granules that need band backfill via the Data Pool Maintenance GUI or by inspecting the EcDIInsertUtilityDPAD.log file. In both cases, the type of error encountered is:

```
ERROR publreg operation encountered a convertEnabled granule with no  
band information, granuleState
```

For each Data Pool granuleId in its input file, the utility will perform the following steps:

1. Validate that the granule is in the hidden Data Pool. The granules can belong to DPL Ingest (isOrderOnly = H) or to OMS (isOrderOnly = Y).
2. Validate that the granule belongs to a convert-enabled ESDT.
3. Validate that the DPL database contains no band information for this granule.
4. Extract the band information from the granule data files and produce a .BandHeader file. This step is performed by invoking an external script (./CUSTOM/utilities/EcDIAdHEGStart). The same script is also used by the new Data Pool Action Driver to create the .BandHeader file during granule registrations. Note: for a multi-file granule, the first file that contains band information will be used.
5. Parse the .BandHeader file and insert the necessary information in the Data Pool database. The .BandHeader file will be removed once it has been parsed.
6. Request the publication of the backfilled granule by inserting a record in the DIInsertActionQueue table in the Data Pool database.
7. Process the next granule in the input file. Note: if an error is encountered during the processing of a granule, the error is logged and the utility continues with processing of the subsequent granules.

4.8.14.1 Using the Data Pool Band Backfill Utility

The Data Pool Band Backfill Utility is started via the following script, from the /usr/ecs/<mode>/CUSTOM/utilities directory:

```
EcBandBackfillUtilityStart -mode <mode> -file <input file>
```

There are two command line parameters that are used in combination with each other. Table 4.8.14-1 provides a description of these parameters.

Table 4.8.14-1. Data Pool Band Backfill Utility Command Line Parameters

Parameter Name	Description
-mode <mode>	Specifies the mode of operation (OPS, TS1, etc.)
-file <input file>	Specifies the full path and file name of the file containing the Data Pool granule IDs of the granules that need to be populated with band information. The file is a flat ASCII file and it contains one Data Pool granuleId per line.

An incorrect command line will result in a ‘usage’ syntax display. The log file for the utility is /usr/ecs/<mode>/CUSTOM/logs/EcDIBandBackfillUtility.log.

4.8.14.2 Data Pool Band Backfill Utility usage examples

Below is an invocation example:

```
1. EcBandBackfillUtilityStart          -mode          OPS          -file
   /home/cmshared/granuleIds.txt
```

Backfills the band information and requests the DPL publication for the granuleIds contained in the specified file. The file contains one Data Pool granuleId per line.

4.8.14.3 Required Operating Environment

The Data Pool Band Backfill Utility will run on a LINUX platform. It shall be installed on the DPL platform as part of the New Data Pool Insert Utility installation.

4.8.14.4 Interfaces and Data Types

Table 4.8.14-2 lists the supporting products that this tool depends upon in order to function properly.

Table 4.8.14-2. Interface Protocols

Product Dependency	Protocols Used	Comments
Data Pool database	SQL	Java JDBC invocation of Stored Procedures.
StoreNext client	Proprietary	Exposes the DPL file system on the DPL platform.

4.8.14.5 Input File Format

One granuleId per line.

4.8.14.6 Configuration File

No special configuration file is needed to run the utility. It uses the same configuration file as the Data Pool Insert Utility (DPIU) and the New Data Pool Insert Utility (NDPIU), namely EcDIInsertUtility.properties.

4.8.14.7 Special Constraints

The mode specific database needs to be up and running and the installation platform need to have access to the Data Pool file system.

4.8.14.8 Outputs

The output of pertinent events is recorded in the /usr/ecs/<mode>logs/EcDIBandBackfillUtility.log log file.

4.8.14.9 Event and Error Messages

Usage errors will be displayed to the screen. Processing error messages are written to the log files.

4.8.14.10 Reports

None

4.8.14.11 Logs

The utility produces log files in the standard log file location. The log file name is EcDIBandBackfillUtility.log. The verbosity of the log file is controlled by the DEBUG_MESSAGES entry in the EcDIInsertUtility configuration file.

4.8.14.12 Recovery

No recovery mechanism is required for this utility. In the event of an interrupted run, the run may be invoked again with the same command-line parameters. Any granules already processed will be detected and not processed again.

4.8.14.13 Database Error Handling

If a database error occurs, the specific error details will be logged. Some database errors are retried internally (i.e. deadlocks), others will cause processing of the current granule to fail and the utility to start work on the next granule in the list.

This page intentionally left blank.

4.8.15 XML Replacement Utility

The XML Replacement Utility (XRU) is an operational support tool used for replacing granule xml metadata files in the ECS inventory. The main role of this utility is to allow the DAAC operator to validate the new metadata file and to inform ECHO of the metadata change. The utility will only replace the granule xml metadata file in the ECS archive. The operator is responsible for making any needed changes to the Inventory database, the DataPool database, and the granule xml metadata file within the DataPool file systems.

4.8.15.1 Using the XML Replacement Utility

The XRU is started by executing the following:

```
$ EcDsAmXruStart <mode> [ -xmlfile <filename> | -xmlmdir <directory> ]
```

The mode parameter is required and either `-xmlfile` or `-xmlmdir` needs to be supplied.

`<mode>`: the mode to run the utility in

`-xmlfile <filename>`: if specified, the utility will perform xml replacement on a single xml file. `<filename>` is the full directory path to the file which has been edited.

`-xmlmdir <directory>`: if specified, the utility will go to the supplied directory path and perform xml replacement on all xml files within the directory

The user running the utility must have read and write privileges in the source and destination (the smallfile archive) directory. The directory must be visible on the host where the utility is run.

4.8.15.2 Required Operating Environment

Linux

4.8.15.3 XML Filename Format

Filename

The input granule xml metadata file must follow the following naming convention:

`<shortname>.<versionid>.<ecsid>.xml`

e.g. MOD29P1D.004.48903.xml, where `<ecsid>` is the dbID in the DsMdGranules table in the Inventory database.

This naming convention is the same naming convention used by the granule xml metadata file in the ECS inventory. The utility assumes that the operator will copy the granule xml metadata file out of ECS and modify the contents for replacement. For any xml files that do not follow the naming convention the replacement will fail and an error will be logged. Any files not ending with an extension of “.xml” will be ignored.

4.8.15.4 Configuration File Format

Table 4.8.15-1 shows the configuration file parameters.

Table 4.8.15-1. Configuration File Parameters

Parameter Name	Description
PROGRAM_ID	The ECS Program ID for the XRU user (SYB_USER)
DB_INVENTORY_HOST	The host for the Inventory database
DB_INVENTORY_PORT	The port number used to connect to the Inventory database
DB_SPATIAL_PORT	The port number used to connect to the sqs server
DB_NAME	The name of the Inventory database
DB_INVENTORY_USERNAME	The username used to connect to and perform queries for the Inventory database
DB_RETRIES	The number of times to retry failed DB operations
DB_SLEEP_SECONDS	The number of seconds between DB operation retries
DEBUG_MESSAGES	The flag that signifies if debug log messages will be logged or not. Valid values are "Y" and "N".
DB_JDBC_CLASS	The JDBC class used for connecting the XRU application to Sybase
DB_TIMEOUT_SECONDS	The number of seconds that a database operation will execute before timing out

4.8.15.5 Outputs

Output will be displayed on the terminal (standard out) where the utility is executed. The output will display one line for each file being processed showing the name of the file and the result for that file; (either "done" or "failed", with "done" meaning successful replacement and "failed" meaning unsuccessful replacement). An example of the output is displayed below:

```
f4dp101{cmshared}114: EcDsAmXruStart DEV04 -xmlmdir /home/cmshared/source
Initializing...
Processing...
AE_Land.086.55515.xml...failed!
AE_Land.086.55516.xml...failed!
AST_L1A.003.48619.xml...done!
MOD29P1D.004.48909.xml...done!

Success: 2, Fail: 2, Total: 4
```

A log file will be generated to log the execution of the utility and any errors that occurred during the xml replacement run. The log file will be located in the mode's log directory with the name, EcDsAmXru.log. Older logs will have a timestamp appended to the end of the log filename.

4.8.16 DPL cleanup orphan/phantom validation (EcDICleanupFilesOnDisk.pl)

EcDICleanupFilesOnDisk.pl provides a mechanism for the ECS Operator to perform validation of the Data Pool disk holdings. It can also be used to remove files from the DataPool directory structure that do not have an associated entry in the Data Pool Database (orphans). In addition it will create output files specifying any granules in the Data Pool Database whose files are missing from the Data Pool disk (phantoms).

An orphan is a file on disk that is not in the DPL Database. It can be:

- a. A file in the public or hidden Data Pool directories that does not have a matching entry in the Data Pool inventory database;
- b. A symbolic QA/PH/browse link that does not match a QA/PH/browse cross reference.
- c. A symbolic hidden order link that is not implied by the presence of a public granule that is also on order (i.e., is in isOrderOnly state 'B').
- d. Browse files with no matching entry in the DIBrowseFile table.

The most likely cause of an orphan file is that the granule or order was cleaned up but that the file or link removal failed for some reason.

A science granule is considered a phantom in the following cases:

- a. There is at least one file entry for such a granule in the Data Pool database DIFile table where the corresponding file does not exist on the Data Pool disks.
- b. A Browse cross reference for a public granule in DIGranuleBrowseXref is not represented by a link in the public directory to the associated browse file.
- c. A QA cross reference for a public granule in DsMdQaGranuleXref is not represented by a link the public science directory to the associated QA file.
- d. A PH cross reference for a public granule in DsMdGranules.ProcessingHistoryId is not represented by a link in the public science directory to the associated PH file.
- e. The granule is a public granule referenced by an order and a hidden link needed to support the order is missing from the hidden directory.

A browse granule is considered a phantom in the following case:

- a. A Browse granule with a Browse File that is missing from the Data Pool disk.

A QA/PH granule is considered a phantom in the following cases:

- a. A public QA/PH granule with a QA/PH File that is missing from the Data Pool disk.
- b. The public QA/PH granule is referenced by an order and a hidden link needed to support the order is missing from the hidden directory.

In addition, a database validation will find any Browse with no associated science granules by default.

If the operator requests that the utility fix files on the Data Pool disk that are missing from the database by removing the files, the utility will also remove files in the temporary directories older than a specified age (MAX_ORPHAN_AGE or maxFileAge) as well as any “DpRecentInserts” files older than 7 days.

Any directories that are made empty due to a cleanup run will be removed.

The utility will exit with a code of 0 for successful validation with no discrepancies, 1 for a failed run caused by an internal error or 2 for a successful validation with discrepancies.

4.8.16.1 EcDlCleanupFilesOnDisk.pl Configuration File

The utility uses a configuration file, EcDlCleanupDataPool.CFG, located in the /usr/ecs/<mode>/CUSTOM/cfg directory. The configuration parameters are stored in a PARAMETER = VALUE format with each parameter/value pair as a separate line entry in the file. The following table describes the configuration parameters which are applicable to EcDlCleanupFilesOnDisk. Table 4.8.16-1 describes the Configuration Parameters.

Table 4.8.16-1. Configuration Parameters (1 of 2)

Parameter Name	Value Description
SYB_USER	Sybase login name for the user of the Data Pool database.
SYB_SQL_SERVER	Name of Sybase SQL Server hosting Data Pool database.
SYB_DBNAME	Name of Data Pool database.
PGM_ID	Program identifier used as seed to generate database password.
NUM_RETRIES	Number of times database operation will be attempted.
SLEEP_SEC	Number of seconds between retries.
MAX_ORPHAN_AGE	Maximum age in days in qualifying a file as an orphan. A file must have an age greater than or equal to this value in order to be considered as an orphaned file. The parameter value must be 3 days or greater.
VALIDATION_OUTPUT_DIR	The standard output directory as configured to be /workingdata/emd/<mode>/DPLCleanup. The value of the -outputDir command line parameter will be appended to this path to determine where the output files will be written.
ORDER_OUTPUT_COLLECTIONS	List of collection groups for which files are stored in the Data Pool file systems, but whose files are not tracked in the Data Pool database by design. The suggested value for this field is “OUTPUTS BRWS”. The MAX_ORDER_AGE parameter applies only to collections in this list.

Table 4.8.16-1. Configuration Parameters (2 of 2)

Parameter Name	Value Description
MAX_ORDER_AGE	A file in a collection group specified in ORDER_OUTPUT_COLLECTIONS must have an age greater than or equal to this value in days in order to be considered as an orphaned file. Suggested value for this field is 15.

4.8.16.2 Using the EcDlCleanupFilesOnDisk Utility

The EcDlCleanupFilesOnDisk utility should be started by the cmshared user. If no parameters are given to the utility it will display the following usage statement:

```
EcDlCleanupFilesOnDisk.pl <mode >
    [-collgroup <"group1 group2" ....>]      -- optional
    [-maxFileAge <age in # of days>]         -- optional
    [-outputDir <outputDir>]                 -- optional
    [-fix]                                    -- optional
    [-debug]                                  -- optional
```

The <mode> parameter is mandatory as the first parameter. If no other parameters are given the utility will validate all of the DataPool for the mode. This may take a long time to run, so the DAAC operator could begin by validation of a selection of collection groups by using the collgroup option and specifying a list of collection groups enclosed by double quotes. The operator may only want to find files on disk which are not in the database if they are older than a certain number of days, in which case he could use the -maxFileAge option. The -fix option can be used if the operator wants to clean up the files on disk that are not referenced in the database. Table 4.8.16-2 describes the Command Line Parameters.

Table 4.8.16-2. Command Line Parameters (1 of 2)

Parameter Name	Required	Description
-collgroup	No	Allows the user to specify a list of collection groups on which to perform validation/cleanup.
-maxFileAge	No	Specifies how old the file must be before considering it as missing from the Data Pool database. If this parameter is not specified the value of MAX_ORPHAN_AGE in the configuration file will be used. The minimum value used will be three days.
-outputDir	No	By default the output files are written to the configuration parameter "VALIDATION_OUTPUT_DIR". If the -outputDir option is specified CleanupFilesOnDisk will create a subdirectory in which to store the output files.

Table 4.8.16-2. Command Line Parameters (2 of 2)

Parameter Name	Required	Description
-fix	No	Allows the user to delete files on disk that are not in the Data Pool Database. It is recommended to not use this option until a validation run has been examined.
-debug	No	This option causes CleanupFilesOnDisk to log extensive information that can be useful when trying to track down validation errors. It will also create two files for each collection, one with a list of files CleanupFilesOnDisk is expecting to find on disk based on what it queried from the database, the other with a list of files actually found on the DataPool disk. These files will be stored in the CUSTOM/temp/DPL directory. These debug files must be cleaned up by the operator.

4.8.16.3 Cleanup Files On Disk command line examples

EcDICleanupFilesOnDisk.pl TS1

This call will validate all collection groups and Browse in the TS1 mode. It will create output files specifying any discrepancies that were found in the default output validation directory.

EcDICleanupFilesOnDisk.pl TS1 – collgroup MOST -maxFileAge 5

This call will only validate the MOST collection group along with its browse cross reference (a link in MOST public directory to the associated browse file) ignoring any files that were modified less than 5 days ago. It will not check any browse files associated with MOST granules.

EcDICleanupFilesOnDisk.pl TS1 – collgroup “MOST BRWS” -maxFileAge 5

This call will validate the MOST collection group ignoring any files that were modified less than 5 days ago. It also validates the Browse holdings in TS1 after checking to see if there are any browse that are not associated with a science file.

EcDICleanupFilesOnDisk.pl TS1 -outputDir mytest

Validates all collection groups and puts any repair output files into a subdirectory called “mytest”.

EcDICleanupFilesOnDisk.pl TS1 -debug

Validates all collection groups and prints extensive information about the validation tests being performed in the log. This includes the sql used to query the Data Pool database for the files in a given collection and the associated find command that is run on the Data Pool disk. In addition the files found in the database and the files found on disk for each collection are written to separate output files in the /usr/ecs/<MODE>/CUSTOM/temp/DPL/ directory.

EcDlCleanupFilesOnDisk.pl TS1 -collgroup "ASTT BRWS" -debug -fix

Validates all collections belonging to the ASTT, BRWS collection groups in the TS1 mode and removes any files found in the collection directories that were not represented in the Data Pool database. Any Browse no longer referenced by a science granule will be removed from the Data Pool database. The Browse files in DlBrowseFile will be compared with the Browse files found using the UNIX find command on the Browse public directory and any files returned by the find command that were not returned by the SQL will be removed. Granules with files in DlBrowseFile that were not found on the disk will be logged to output files. Due to the use of the -debug option extensive logging and debug files will be available after the run.

4.8.16.4 Required Operating Environment

The DataPool Cleanup Files utility will run on a Linux platform. It requires the GNU find command which is installed with RedHat version 5.

4.8.16.5 Databases

Table 4.8.16-3 lists the supporting products that this tool depends upon in order to function properly.

Table 4.8.16-3. Product Dependencies

Product Dependency	Protocols Used	Comments
DataPool Database	SQL	Via SQL server machines

4.8.16.6 Special Constraints

This utility will not process any files systems that are marked as unavailable in the DataPool database.

4.8.16.7 Output Files Naming Convention

The output files will be written to a subdirectory specified by the -outputDir command line parameter of the directory configured in the VALIDATION_OUTPUT_DIR configuration parameter.

The utility will create the following three output files that could be used as input for the RestoreOLAFromTape utility.

a.GranulesMissingFiles_dplids_RepairByRestoreOLAFromTape.<processid>.<yyyymmddhhmmss>

-- a list of the Data Pool granuleIds whose files are missing from the Data Pool disk.

For example:

30176
30168
30177

b.GranuleFilesNotOnDisk_files_RepairByRestoreOLAFromTape.<processid>.<yyyymmddhhmss>

-- a list of files with full paths that are missing from the Data Pool disk.

For example:

```
/datapool/DEV04/user/FS1/.orderdata/AMSALIMeVZZf/AE_Land.086PjFMBJRv/2006.07.14/tny_2sci_2browse.1187370525.76499.RGEN.hdf
```

```
/datapool/DEV04/user/FS1/.orderdata/AMSALIMeVZZf/AE_Land.086PjFMBJRv/2006.07.14/tny_2sci_2browse.1190311192.11894.RGEN.hdf.xml
```

c.BrowseMissingFiles_dplids_RepairByRestoreOLAFromTape.<processid>.<yyyymmddhhmss>

-- a list of browse granule ids whose files are missing from the Data Pool disk.

For example:

```
80584
80587
78345
```

The utility will also create the following two files:

a. BrowseWithNoScienceGranule_dplids.<processid>.<yyyymmddhhmss>

--a list of browseIds with no cross-references to any public science granules.

For example:

```
303671
303677
303678
```

b. **FilesNotInDatabase_files.<processid>.<yyyymmddhhmss>**

--a list of the orphaned files with full path names,

For example:

```
/datapool/DEV04/user/FS1/LSR7/MODPTQKM.086/2005.11.06/MODPTQKM.A2005310.h00v09.004.2005312161035.hdf.0220.1193161669.56934.RGEN.hdf
```

The utility will create the following two additional output files in the /usr/ecs/<MODE>/CUSTOM/temp/DPL directory if -debug is specified:

a. **FilesOnDisk.<directoryPath>.processId.yyyymmddhhmss:** a list of files found from the Data Pool disk for a given collection.

For example:

FilesOnDisk.DAP.001.1443.20090109120027 could contain the following file:
/datapool/DEV04/user/FS1/.orderdata/GLASChabExaP/GLA0PRAP.001BKugFfta/2006.09.30/P
2061984AAAAAAAAAAAAAAAAA06273091132100_2.PDS

b. FilesInDb.<directoryPath>.processId.yyyymmddhhmmss: a list of files found from the Data Pool inventory for a given collection.

For example:

FilesInDb.MODPTQKM.086.1443.20090109120034 could contain the following file:

/datapool/DEV04/user/FS1/LSR7/MODPTQKM.086/2005.11.06/MODPTQKM.A2005310.h00v
09.004.2005312161035.hdf.0220.1193162576.93626.RGEN.hdf

FilesInDb.MODPTQKM.086YkZDjSTj.1443.20090109120035 could contain the following file:

/datapool/DEV04/user/FS1/.orderdata/LSR7VEkFZrNF/MODPTQKM.086YkZDjSTj/2005.11.0
6/MODPTQKM.A2005310.h00v09.004.2005312161035.hdf.0220.1193266902.75079.RGEN.hd
f

4.8.16.8 Event and Error Messages

Usage errors will be displayed to the screen. Processing error messages are written to the log.

4.8.16.9 Logs

A single log named EcDICleanupFilesOnDisk.log will keep track of the start and stop of each run. It will contain updates as it processes each collection including the number of error files found and the amount of disk space occupied by these files. Extensive logging can be obtained with the `-debug` option.

4.8.16.10 Recovery

This utility does not provide recovery for abnormally terminated runs. Rerunning the utility with the same parameters will complete any aborted run.

4.8.16.11 Sybase Error Handling

All Data Pool Cleanup components will attempt to deal with Sybase errors gracefully, usually by retrying the query depending on `NUM_RETRIES` set in the configuration file. If a query cannot be completed after several retries, the utility will immediately terminate with an exit code of 1.

If a Sybase error occurs, the actual Sybase error string will display on the screen and in the log.

This page intentionally left blank.

4.8.17 The DataPool Cleanup Granules Utility

The DataPool Cleanup Granules Utility (EcDICleanupGranules.pl) provides a mechanism by which the ECS Operations Staff can remove granules from the DataPool. It will remove the files and database entries associated with the specified granules from the DataPool. The DAAC staff can specify a list of granules on the command line, or through an input file. The utility has the ability find all granules that were removed from the archive and delete them from the DataPool. The utility can be used to remove expired non-ECS granules from the DataPool. All deletions of ECS granules will be exported to ECHO, and the utility has the ability to export deletions to ECHO before the DAAC removes the granules. If the utility is not able to remove the granule because it is on order or the file system for the granule is unavailable or there is a lock on the granule, deletion will be postponed until the next run.

4.8.17.1 The DataPool Cleanup Granules Utility Configuration File

The utility uses the shared configuration file EcDICleanupDataPool.CFG, located in the /usr/ecs/<mode>/CUSTOM/cfg directory. The configuration parameters are stored in a PARAMETER = VALUE format with each parameter/value pair as a separate line entry in the file. The following table describes the configuration parameters applicable to Cleanup Granules.

Table 4.8.17-1. Configuration Parameters

Parameter Name	Value Description
SYB_USER	Sybase login name for the user of the Data Pool database.
SYB_SQL_SERVER	Name of Sybase SQL Server hosting Data Pool database.
SYB_DBNAME	Name of Data Pool database.
PGM_ID	Program identifier used as seed to generate database password.
DEFAULT_LIMIT	Default retention priority limit. This option is used when cleaning up expired NON-ECS granules (with the – expired command line parameter). Each granule can be assigned a retentionPriorityLimit in the DIGranuleExpirationPriority table. If the limit assigned to a granule is higher than the limit used by Cleanup the granule will not be cleaned up. The user has the option to override this limit by using the -limit command line parameter. This is usually set to 200, so that granules with a priority limit greater than 200 will not be removed from the Data Pool.
NUM_RETRIES	Number of times database operation will be attempted.
SLEEP_SEC	Number of seconds between database retries.
BATCH_SIZE	Number of granules per batch, suggested value for this field is 500.

4.8.17.2 Using the DataPool Cleanup Granules Utility

Invocation of `EcDICleanupGranules.pl` should be performed by the `cmshared` user. The first parameter is the operational mode that contains the granules that are to be cleaned up. It is required. If the utility is run without any other parameters it will clean up all granules left by any previous run. To specify new granules to be cleaned up, the following five parameters can be used:

- `-file <filename1 filename2 ...>`
- `-geoidfile <filename1 filename2>`
- `-grans <gran1 gran2 gran3 ...>`
- `-ecsgrandel`
- `-expired`

To limit possible database contention between the cleanup run and other ECS services the command line parameter `-batchsize <number of granules per batch>` can be used to force Cleanup Granules to break up all the granules it will cleanup into sets with the specified size. If this parameter is not specified the value of `BATCH_SIZE` in the configuration file will be used.

If the DAAC Operators would like to inform ECHO that the granules they are about to clean up will no longer be available they can run with the `-exportonly` option. This will postpone the cleanup of the specified granules until the next run of `EcDICleanupGranules.pl`.

Non-ECS granules may have expiration dates associated with them. To clean up Non-ECS granules in the DataPool that have expired, the Cleanup Granules Utility offers the DAAC operators a `-expired` option. This option can be qualified with the `-offset` option to specify the number of hours after midnight of the previous day in which to consider a granule expired. By passing a negative number to the `-offset` option the operator can specify an expiration time before yesterday at midnight. In addition granules can be excluded from expiration regardless of their expiration date by setting their `retentionPriority` to be larger than the `-limit` option, which defaults to 200. Finally one can limit the cleanup of expired granules by their theme using the option `-theme`.

In extreme error cases in which the utility fails to start or fails to process new deletions the command line option `-norecovery` may be used which will remove all pending granules that should be cleaned up. Use of this option is not suggested and may cause Online Archive validation errors.

The usage statement for the DataPool Cleanup Granules Utility is:

```
EcDICleanupGranules.pl <mode>
    [-norecovery]
    [-file <input1 input2 ... inputn>]
    [-grans <granId1 granId2 .. granIdn>]
    [-geoidfile <input1 input2 .. inputn>]
    [-ecsgrandel]
```

[-expired]
 [-offset <# of hours>]
 [-limit <priority limit>]
 [-theme <theme1 theme2 .. themen>]
 [-themexref <theme1 theme2 .. themen>]
 [-batchsize <batch size>]
 [-exportonly]

Table 4.8.17-2. Command Line Parameters (1 of 2)

Parameter Name	Required	Description
-file	No	Full path of File name with a list of DPL granule ids as input
-grans	No	A list of DPL granule ids separated by spaces and enclosed in double quotes.
-geoidfile	No	Specifies the full path name of the file containing geoids which are a combination of science type, esdt short name and version id and ECS Archive Inventory Management (AIM) database id. Granules in this file whose ECS id match those in the data pool are candidates for data pool cleanup if specified by this option. May not be used in conjunction with any other options other than the <i>noprompt</i> option. Note that the geoid file can contain science granules as well as non-science granule because the AIM Granule Deletion Utility may delete these types of granules. The input value for this parameter is logically defined to be the output of any AIM phase 1 (EcDsBulkDelete.pl) granule deletion run. This will cause the Data Pool cleanup utility to clean up any AIM granules found in the geoid input file to be removed from the Data Pool database.
-ecsgrandel	No	Indicates that only granules removed in the ECS system from the AIM Inventory database will be removed from the data pool if they exist. No other cleanup will occur.
-expired	No	Cleanup non-ECS granules by expiration date and retention priority
-offset	No	Use only with the <i>-expired</i> option. Specifies hours before (negative) or after (positive) midnight of the previous day from which to delete. Defaults to zero. (Some examples: -offset 5 would delete all granules which had expired as of 5 AM of the current day; -offset -5 would delete all granules which had expired as of 7 PM yesterday -offset 72 would delete all granules which will be expired before midnight two days from now.

Table 4.8.17-2. Command Line Parameters (2 of 2)

Parameter Name	Required	Description
-limit	No	Use only with the <code>-expired</code> option. Specifies limiting value used for determining which granules will be deleted. Will delete all granules with priority less than or equal to the specified limit. Must be within the range 1–255, 1 being the lowest priority and 255 being the highest priority. Defaults to value specified in configuration file.
-theme	No	Use only with <code>-expired</code> option. Specifies the name of a theme for which cleanup is to be performed. The Cleanup Utility will clean up granules that would otherwise qualify for cleanup only if the granules are associated with that theme, and remove the granules entirely if they are not associated with any other theme, otherwise only remove the cross references with that theme. The theme name must be enclosed in quotes ("").
-exportonly	No	Only export granules that are going to be deleted to ECHO without actually cleaning them up. They will be cleaned up in the next run of <code>EcDlCleanupGranules.pl</code> .
-batchsize	No	Process cleanup by batch files. Recommended batchSize for large cleanups is 500.
-norecovery	No	Do not recover unprocessed DPL granules, and ECS granules that were left from the failure of a previous program execution. Also remove DPL granules and ECS granules that were not deleted because they were on order or file system was unavailable. It is recommended not to use this parameter.

4.8.17.3 Examples:

Cleanup using a file containing a list of Data Pool granule ids

```
EcDlCleanupGranules.pl TS1 -file /home/abc/myfile1 /tmp/myfile2 -batchSize 500
```

Cleanup using a list of Data Pool granule ids

```
EcDlCleanupGranules.pl TS1 -grans 30987 90876
```

Export granules that will be deleted to ECHO only:

```
EcDlCleanupGranules.pl TS1 -file /home/abc/dplidfile
                        -grans 30987 90876
                        -ecsgrandel
                        -exportOnly
```

Cleanup nonECS granules with -expired option

```
EcDlCleanupGranules.pl TS1 -expired -offset 5 -limit 200 -theme test1 test2
```

Propagate AIM granule deletions to the Data Pool

```
EcDlCleanupGranules.pl TS1 -ecsgrandel -batchSize 500
```

Cleanup using a file containing a list of geoids

```
EcDlCleanupGranules.pl TS1 -geoidfile /home/abc/geoidfile1 /tmp/geoidfile2
```

4.8.17.4. Outputs

The EcDlCleanupGranules.pl script does not produce output files.

4.8.17.5. Recovery

The EcDlCleanupGranules.pl script provides a capability to recover from an execution failure caused by situations such as system faults or database errors leaving all or some of the deletes unprocessed. The script will detect such failures upon the next run and continue processing the deletes that were left unprocessed in the previous run. Recovery will proceed to prevent the Data Pool inventory and disk files from existing in a corrupted state.

4.8.17.6 Sybase Error Handling

All CleanupGranules components will attempt to deal with Sybase errors gracefully, usually by retrying the query depending on NUM_RETRIES set in the configuration file. If a query cannot be completed after several retries, the utility will immediately terminate with an exit code of 1.

If a Sybase error occurs, the actual Sybase error string will display on the screen and in the log.

4.8.17.7 Event and Error Messages

Usage errors will be displayed to the screen. Processing error messages are written to the log files; some error messages will displayed to the screen as well.

The log file is named EcDlCleanupGranules.log, and is stored in the /usr/ecs//<mode>/CUSTOM/logs directory.

This page intentionally left blank.

4.8.18 Line Checker Utility (EcDILinkCheck.ksh)

The EcDILinkCheck.ksh script is used to find or delete ‘broken’ symbolic links, i.e., links that point to files that do not exist. The Data Pool contains three varieties of links. The public ESDT directories contain links that point to browse files associated with science granules. The hidden ESDT directories contain links for ordered granules that point to public science files. The FTP pull directory that is used by OMS contains links to files/links in the hidden ESDT directories.

Broken links are most likely to occur due to a failure to remove the link at the appropriate time. It is possible for the links to be created before the file that they point to is in place. It may be prudent, therefore, for DAAC staff to verify that the reported links are indeed broken and then remove the broken links manually via a UNIX ‘rm’ command. Running the utility with the –fix option would be useful when the system encounters a situation in which there are too many broken links to manually verify and remove.

Another option is to run the utility against a single public or private collection directory. This will reduce the number of broken links discovered and also reduce the amount of time the utility takes to complete, allowing the operator to validation links in real time.

The utility will log all information to the file “./logs/EcDILinkCheck.log” and output all broken links.

Table 4.8.18-1. Command Line Parameters

Parameter Name	Required	Description
<DIRECTORY_TO_START_CHECK>	Yes	Allows the user to specify the directory on which to perform the broken link check.
-fix	No	Allows the user to remove the invalid links.

4.8.18.1 Using the EcDILinkCheck.ksh Utility

EcDILinkCheck.ksh <DIRECTORY_TO_START_CHECK> [-fix]

Examples:

EcDILinkCheck.ksh /datapool/TS2/ –fix

The utility will remove all broken links in the TS2 Data Pool.

EcDILinkCheck.ksh /datapool/TS2/user/FS1/MOLA > MOLAbrokenbrowselinks.txt

The utility will search for broken browse links for the MOLA collection group.

EcDILinkCheck.ksh

**/datapool/TS2/user/FS1/.orderdata/<encryptedASTT>/<encryptedAST_L1B.003> >
asterlinks.txt**

The utility will search for broken order links for AST_L1B version 3 granules.

EcDILinkCheck.ksh /datapool/TS2/user/FS1/PullDir> brokenPullLinks.txt

The utility will search for broken links in the FTP Pull directory used by OMS.

4.8.18.2 DataPool Link Check Configuration File

None

4.8.18.3 DataPool Link Check Log Files

EcDILinkCheck.ksh will create a file named EcDILinkCheck.log in the ../logs directory.

4.8.18.4 DataPool Link Check Output Files

EcDILinkCheck.ksh will output to stdout the full path names of all the links it determines are broken. It is suggested that the DAAC operator pipe stdout to an output file of their choosing.

4.8.19 AIM Tape Archive Consistency Checking Utility

The **EcDsAmArchiveCheckUtility** checks the contents of the Volume Group directories defined in the Inventory Database against the list of files recorded in the Inventory Database. These data files are typically stored on tape, but may also be stored on disk. The utility will check all granule types (Science, Browse, Production History, QA, and Delivered Algorithm Package). The utility verifies the name, location, and size of the files in the Inventory. It does not verify checksums.

The utility also includes a check of the XML Archive by default; this option is appropriate for processing a small volume of data, such as a few days, but should be avoided if you are testing a large amount of granules because it adds significant processing time. The **EcDsCheckXMLArchive.pl** utility is much faster for checking the XML Archive. There is a command line option for the **EcDsAmArchiveCheckUtility** to turn off the XML archive check (-nx).

This utility uses the standard ECS volume group rules for determining what granules are mapped to a given volume group. The rules are based upon comparing the metadata of a granule to the attributes of the volume group stored in the **DsStVolumeGroup** table within the Inventory database. Within these rules, the **ShortName** and **VersionID** of the granule are compared to the volume group's **VersionedDataType** attribute. Next, the **insertTime** of the granule must be between the **VolumeStartDate** and **VolumeEndDate** for the volume group. Finally, if the volume group has a value for **SelectionDate**, it is compared to the granule's **BeginningDateTime** to determine if the granule is part of a forward or reprocessing chain (this applies to science granules only). If the **BeginningDateTime** is less than the **SelectionDate**, the granule is part of a reprocessing chain and should be located in the volume group that matches the above conditions and has an "R" appended to the **VersionedDataType**. If the **BeginningDateTime** is greater than or equal to the **SelectionDate**, the granule is part of a forward processing chain and will be located in the volume group without an "R" appended to the **VersionedDataType** attribute.

4.8.19.1 EcDsAmArhiveCheckUtility – Command line options

The utility has the following command line options:

```
EcDsAmArchiveCheckUtilityStart <MODE> [-ar ArchiveRootPath] [<directories to check>] [-s] [-o] [-lo (Output Directory)] [-nx]
```

```
<directories to check> = -d <date_range> | -e { <ESDT> ... }  
| -a  
| -v <volume_group_path>  
| -vs <starting_volume_group_path>
```

Example:

```
EcDsAmArchiveCheckUtilityStart OPS -ar /stornext/snfs1/ -e "AST_L1B.001" -s -o -lo -nx
```

Table 4.8.19-1 shows the command line parameters for the **EcDsAmArchiveCheckUtility**.

**Table 4.8.19-1. Command Line Parameters of the EcDsAmArchiveCheckUtility
(1 of 2)**

Parameter Name	Mandatory	Description
<MODE>	Yes	The mode to be processed.
-ar ArchiveRootPath	No	The directory of the archive root path. If this option is not present, the utility will check all the volume groups. If this option is present, then the utility will check only the volume groups whose VolumeGroupPath match <ArchiveRootPath>/<MODE>.
-d	No	Select only those granules whose insert time falls within the supplied date range For example: -d "Mar 13 2002 - Mar 22 2002" or -d "Mar 3 2002 1:23 PM - Mar 22 2002 15:51 PM If you omit the end date, the utility will use the current date (day) as a default end date. With this option the utility will report database entries with missing files but it will not report files with missing database records.
-e	No	Only check granules whose ShortName and VersionID are specified by the provided list of ESDTs. See the notes about this option described later in this section.
-a	No	Only check "active" volume groups, i.e., directories that are still open for receiving files. These volume groups are determined by the VolumeEndDate being set to null in the DsStVolumeGroup table.
-v	No	Process the supplied volume group path. This should be a fully qualified path starting at the root directory.
-vs	No	Allows the operator to provide a fully qualified path where the utility should begin processing. The supplied volume group and all volume groups that were created after the starting volume group will be processed. This allows the operator to process all volume groups created since a specific time. In addition, the -vs option can alternatively be passed a VolumeGroupID from the DsStVolumeGroup table instead of a fully qualified path. The volume groups that are skipped are also recorded in the output file.
-s	No	Saves a list of all files processed in a file with the name <MODE>.YYYY.MM.DD_HH:MM:SS_<HOST>.dbOut.
-o	No	Output a list of files that match, to a file with the naming convention <MODE>.yyyy.mm.dd_hh:mm:ss_<host>.ok. This should be used with caution as the list could be very large.

**Table 4.8.19-1. Command Line Parameters of the EcDsAmArchiveCheckUtility
(2 of 2)**

Parameter Name	Mandatory	Description
-lo (OutputFilePath)	No	If the option is not present, the utility creates all output files in current working directory where the utility was executed. If the option is present but without being followed by 'OutputFilePath', the utility creates all output files in the log directory for the MODE. If the option is present with an 'OutputFilePath', the utility creates all output files in the directory which 'OutputFilePath' specifies.
-nx	No	Suppress the consistency checking for the XML archive. This allows the utility to run much faster when processing large volumes of data. The EcDsCheckXMLArchive.pl utility can be used to specifically check the XML archive.

If the operator doesn't provide a specification of which volume groups (directories) to check using one of the options (-e, -d -a, -v, or -vs), then the utility will check all volume groups.

The operator will be prompted to enter the Sybase DB Server – for the DAACs this should be

< n | e | l >4dbl03_svr.

Next, the name of the AIM Inventory database is displayed and the operator is prompted to accept the entry by entering the letter A and pressing the enter key or to change the entry by pressing the letter C and the enter key.

Next, the operator is asked to enter a Sybase login name and password; this login must have access to the AIM Inventory (EcInDb) database.

Next, the operator is prompted for an “ArchiveRoot.” Enter the starting part of the volume group path that is to be searched. Typically this will be something like “/stornext/snfs1” which reflects the path up to but not including the <MODE> component of the directory path.

If the operator uses one of the options (-e, -d -v, or -vs) a list of volume groups to be processed will be presented and the operator is asked to accept it by pressing “y” and enter. If operator doesn't use one of the options to limit volume groups to be processed or uses the -a option, this step is omitted.

Notes about using the -e option:

The -e {ESDT ...} option will check all ESDTs in the provided list, note, the {} should not be entered on the command line. If the Version ID for the ESDT is omitted, the program will check all granules whose short name match the passed in short name. Wild cards may also be provided in the ESDT list.

For example, to process all ESDTs that begin with "AST", you would run the program with the following option and parameter: `-e "AST*"` (this will cause ALL versions of every ESDT that begins with "AST" to be processed).

When no "." is present in the wildcard, the program assumes that the supplied argument is a ShortName only, and processes all versions of that particular ESDT. So supplying `-e "MO*3"` will cause the program to search for a VersionedDataType that matches the following pattern `MO%3.%`. If you really wanted to search for all version 3 ESDTs that start with "MO", you would supply the following `-e "MO*.003"`.

Any combination of absolute strings or wild cards can be used with the `-e` option.

For example: `-e "AST*" "MOOD" "A*L1*.002" "MOOO.001"` will cause the processing of all versions of ESDTs that begin with "AST", all versions of ESDTs that have the short name "MOOD", all version 2 ESDTs that begin with "A" and have an "L1" substring, and finally the ESDT MOO0.001.

Another example would be to process only version 2 ESDTs using the option `-e "*.002"`.

When the `-e` option is used, the utility will prompt for confirmation after printing the list of VolumeGroups that correspond to the list of ESDTs that are being used. When using wild card characters, you must enclose the pattern in quotes in order to prevent the UNIX system from interpreting the pattern as a file-matching pattern. The Utility translates UNIX regular expressions into SQL regular expressions, but it leaves SQL regular expressions in the format passed in via the command line. This means that you may use SQL regular expressions on the command line and that pattern will be used when querying the DsStVolumeGroup table to find the correct volume groups. For example: `-e "[^Browse]"` will get all VersionedDataTypes that don't have the Browse ShortName.

Also note that upon execution, the utility removes all pre-existing output files that have the identical time stamp as the output files for the current run in order to "clean up" after a previous run that had errors. So when the utility is run consecutively within the same minute, it assumes that the previous runs in that minute were erroneous.

4.8.19.2 EcDsAmArchiveCheckUtility – Outputs

As the utility runs, it displays a summary of each volume group it processes to the console. In addition, it creates several report files. All output files for the EcDsAmArchiveCheckUtility have the following naming convention:

`<MODE>.yyyy.mm.dd_hh:mm:ss_<host>.<fileExtension>`

where yyyy, mm, dd, hh, mm, and ss describe the current year, month, day of month, hour, minute, and second. The output files and logs are produced in the current working directory of the script or in the standard logs directory if the `“-lo”` option is provided on the command line.

The following report files are produced:

FileName	Contents
<code><MODE>.yyyy.mm.dd_hh:mm:ss_<host>.dupGrans</code>	Duplicates, triplicates, etc.

<MODE>.yyyy.mm.dd_hh:mm:ss_<host>.wrongSize	Granules with non-matching file size
<MODE>.yyyy.mm.dd_hh:mm:ss_<host>.notInArch	Granules missing from archive
<MODE>.yyyy.mm.dd_hh:mm:ss_<host>.notInDb	Files missing from the AIM database
<MODE>.yyyy.mm.dd_hh:mm:ss_<host>.InDbDFAed	Files in the archive but DFAed.

Separate reports are generated to facilitate identification and cleanup of each potential case. Grouping all errors into a single report would potentially make the report too large to easily use.

Files in the archive that are missing database records are sometimes referred to as “orphans” and are usually the result of a Granule Deletion failure. Database records that are missing files in the archive (“phantoms”) or that have file size mismatches are a much more serious issue and most likely represent an error during Ingest or a corruption of the file system.

In addition to the above report files, the file:

<MODE>.yyyy.mm.dd_hh:mm:ss_<host>.outputSummary

contains a summary of the total execution time of the program, the execution time of each major component of the program, and the processing speed (# db files/second and # archive files/second) of the program. Additionally this file contains the following information for each volume group processed:

- the volume group path
- the number of duplicate granules found
- the number of files with incorrect size
- the number of files not in the archive
- the number of files not in the database
- number of files in the archive that are DFA’ed in the database
- the number of XML files not in the archive
- the number of XML files not in the database
- Total number of files in the archive
- Total number of files in the database

For debugging purposes the following files are also created.

FileName	Contents
<MODE>.yyyy.mm.dd_hh:mm:ss_<host>.dbQuery	All database queries used
<MODE>.yyyy.mm.dd_hh:mm:ss_<host>.VGOut	List of each volume group processed
<MODE>.yyyy.mm.dd_hh:mm:ss_<host>.readDirOut	List of files in each volume group
<MODE>.yyyy.mm.dd_hh:mm:ss_<host>.dbOut	List of files from the AIM database
<MODE>.yyyy.mm.dd_hh:mm:ss_<host>.ok	List of files matched successfully

The output format is similar for each file. Typically each report lists each volume group path as it is being processed. After the volume group path, a tabular list of information is presented that contains the information needed to research the problem. This information will typically include granule ID, ShortName, VersionID, the “internalFileName” used by the file, and possibly the file size. The exact output depends on the report file. The information is presented in human readable form.

Special Notes:

If this utility is executed while ECS ingest is occurring, then the reports on “active” volume groups may contain spurious entries in the "notInDb" file. This is because files may have been copied into the archive but not yet entered into the Inventory database. Also note that DeleteFromArchive is only applicable to “science” granules (i.e.: not Browse, QA, PH, and DAP). The non-science granules should always be version 001 and the SelectionDate is not supported.

4.8.19.3 Configuration File Format – EcDsAmArchiveCheck.CFG

The configuration file contains vital details about how the utility will operate. The utility will exit immediately if a configuration file is not available. The file is a plain text ASCII file and has the following format as shown in Table 4.8.19-2.

Table 4.8.19-2. Individual Configuration Parameters

Parameter Name	Description
AIM_DB_PGMID	Sybase connectivity, the ID (16000008) is used to decrypt the DB password based on ECS standards
SYBASE_SERVER	Sybase connectivity, the name of SYBASE data server
AIM_DB_USER_ID	Sybase connectivity, the user name (EcDsAmArchiveCheckUtility) used to login to the SYBASE data server, AIM database
AIM_DB	Sybase connectivity, the database name for the AIM database without the MODE extension

4.8.20 EcDsCheckXMLArchive.pl

The “EcDsCheckXMLArchive.pl” utility checks the entire contents of the XML Archive against the list of XML metadata files contained in the DsMdXMLFile / DsMdXMLPath tables within the Inventory database. Note that XML metadata files are only stored for science granules.

4.8.20.1 EcDsCheckXMLArchive.pl Command line options

The command line parameters are:

```
EcDsCheckXMLArchive.pl -mode <MODE>
                        -user <db user name>
                        -server <Sybase server>
                        -database <Inventory database name>
                        [ -debug <debug level> ]
                        [ -log <log file name> ]
```

The utility will prompt the operator for a value if a required parameter is not provided on the command line.

Table 4.8.20-1 shows the command line parameters for EcDsCheckXMLArchive.

Table 4.8.20-1. Command Line Parameters of the EcDsCheckXMLArchive.pl

Parameter Name	Mandatory	Description
-mode	Yes	The mode to be processed.
-user	Yes	The Sybase login for the utility to use to connect to the database.
-server	Yes	The name of the Sybase server that contains the Inventory database.
-database	Yes	The name of the Inventory database, this should include the mode suffix if applicable. For example, to check XML files in the TS1 mode, provide “EclnDb_TS1.”
-debug	No	This option controls the amount of information written to the log files. The default is 1 which will cause the utility to write a couple of lines of information for each XML directory it processes. The entries will list the actual path, the number of database entries for the path, the number of files in the directory, and the number of files that were matched. Setting this option to 3 will cause the utility to log each file that is checked. Debug level 2 is currently not implemented but is a placeholder for future use.
-log	No	The log file will be written to the logs directory for the MODE and by default be named using ECS standard naming conventions. This option allows the operator to name the log file using non-standard ECS naming conventions.

4.8.20.2 EcDsCheckXMLArchive.pl Outputs

The utility will write progress information to the console as it runs. This progress information is also written to the associated log file within the log directory. The utility will create 2 report files. Both report files begin with a summary containing the report file name and mode that was processed. The report file name contains the date and time when the report was generated.

The first file, `MissingXMLArchive.report.YYYY.MM.DD`, reports missing XML files within the XML archive. Items in this report correspond to entries in the `DsMdXMLFile` table that are missing files in the XML archive. The directory where the file should be located is indicated by the `pathId` column; this is stored in the `DsMdXMLFile` table and it refers to the `dbID` within the `DsMdXMLPath` table. The body of this report file uses the following format:

```
Missing XML files in the XML Archive: <pathId> <the fully qualified path where the file should be>
GranuleId      XML File
<dbID>         <XML File Name from DsMdXMLFile>
```

The second report, `MissingXMLEntryInDB.report.YYYY.MM.DD` lists files found within an XML directory that do not contain an entry in the Inventory database. The body of this report contains the following format:

```
Missing XML file entries in DB: <pathId> <the fully qualified path where the file was found>
GranuleId      XML File
<dbID>         <XML File Name from DsMdXMLFile>
```

Note, the above `GranuleId` is determined by parsing the file name of the “orphaned” file. If the file is not an XML metadata file created by the ECS system, this value may look confusing and will not actually reflect a valid ECS granule.

4.8.21 The Data Pool Cloud Cover Utility script

The command line script enables the DAAC to correct the cloud cover (CC) information stored in the DPL database for public science granules, based on the cloud cover source that is associated to the desired collection via the DPL Maintenance GUI.

The script also modifies the 7.22 cloud cover derivation for granules that have multiple measured parameters and different associated cloud cover values for those parameters. The new derivation chooses the maximum cloud cover value and associates it to the granules as opposed to the old derivation that used the first cloud cover value in the granule XML file for the association.

DAAC operations will determine when the best time to run it is and what collections they want to run it for.

4.8.21.1 Command line invocation

The cloud cover utility script command line syntax is presented below:

```
EcDICloudCoverUtilityStart <MODE> -operation <correct|remove|populate|repopulate> -collection <ShortName.VersionId> [-recovery no]
```

-operation correct: used to correct core metadata values for a given collection. This should run for collections that are correctly configured but the NDPIU used the first value instead of the MAX value (these are the pre 7.23 code collections that use core metadata as the CC source and have multiple measured parameters). It will only update granules in the specified collection if they have multiple parameters. Part of the 7.23 DB patch the maximum DPL granuleId is saved in a CloudCoverUtility table (DICcuRequests) that will be used by the "correct" operation to determine where to stop corrections for the current collection. This is needed in order to allow the DAACs to run the script at any time after the 7.23 deployment and avoid re-computing the CC values for granules that have the correctly derived value for the CC, due to the 7.23 NDPIU cloud cover derivation changes. If the utility does not find the maximum granuleId that is specified in the table (granule has been deleted in the meantime) it should use the next higher granuleId as the stop value. In theory, the `-operation correct` for a given collection should only run once, after the 7.23 operations deployment.

-operation remove: used to completely remove the DPL cloud cover for a given collection. It works on both PSA and core metadata sources. It removes all records in DIGrCloudCover for a given collection. DAAC operator must first remove the CC configuration for the collection via the DPL Maint GUI (set the cloud cover source to "NONE"), stop DPAD, wait for all publications to complete, restart DPAD and then start script with "-operation remove". NDPIU does retrieve the CC source from the DB for each publication so the new CC configuration applies to all granules that are dispatched for publication after the CC configuration change took place. We don't need to suspend publications for the collection while the removal takes place because for all granules that are dispatched after the CC configuration change took place have no info inserted in DIGrCloudCover. The operator can re-run the script after the first run completed in order to be 100% sure that no granules are left in the DIGrCloudCover table for the specified collection.

-operation populate: used for a collection that had no cloud cover configured at all and the DAAC now wishes to have cloud cover available for web drill-down. It works for both PSA and core metadata sources. If the CC source is a PSA, the XML file will be parsed to determine the correct CC value. It is assumed that before the "populate" runs, the correct cloud cover configuration is entered for the collection using the DPL Maint GUI. No explicit granuleId stop value is needed in this mode, we should work on public granules from the specified collection that have no record in the DIGrCloudCover. Sequence: operator configures CC using maint GUI, starts script with "-operation populate". We do not need to suspend/unsuspend at all since the script will only populate public granules from the chosen collection than have no records in DIGrCloudCover. The granules that are being ingested after the DPL Main GUI configuration change is saved will have the correct cloud cover info.

-operation repopulate: used for a collection that was incorrectly configured. It works for both PSA and core metadata sources. It is in fact a remove, followed by a populate operation. If the CC source is a PSA, the XML file will be parsed to determine the correct CC value for the populate part. DPL publications for the collection are suspended programmatically (via DICollections.moveFlag = Y, which causes DPAD to suspend publications for the duration of the period while the flag is set to Y) from the time the removal starts up to the time when the populate starts, when we set moveFlag = N. The suspension is needed to make sure that we won't remove CC records parameters for granules that are published after the configuration change took place. No explicit granuleId stop value is needed, we should work on all public granules from the specified collection that are identified when the utility is run and have no records in the populate step. Operator sequence: correctly configure CC through the DPL Maintenance GUI, start script with "-operation repopulate".

-recovery no: This is optional, and by default, the program will try to recover from the previous incomplete run. There should be only one instance of the utility can be run at a time. The program ensures the previous command is complete before the operator can start a new one. If the previous run wasn't completed, the operator must run the same command again to allow the run to finish, unless the operator aborts the incomplete run and starts a new run by providing "-recovery no" on the command line.

4.8.21.2 Configuration File for Data Pool Cloud Cover Utility Script

The Data Pool Cloud Cover Utility script uses a configuration file containing details about how to connect to Sybase, the number of threads that are used concurrently to perform the requested cloud cover operation and the level of verbosity in the log file. The file *EcDICloudCoverUtility.properties* contains the configuration parameters for the Data Pool Cloud Cover Utility script. The configuration file must be a plain text ASCII file, which has the following format:

```
PGM_ID=10000033
SYBASE_HOST=<string>
SYBASE_PORT=<integer>
SYBASE_DPL_DBNAME=<string>
SYBASE_USER=EcDICloudCoverUtility
SYBASE_JDBC_DRIVER_CLASS=com.sybase.jdbc3.jdbc.SybDriver
DB_RETRIES=<integer>
```

```

DB_SLEEPSECONDS=<integer>
DB_BATCH_SIZE=<integer>
SQL_TIMEOUT_SECONDS=<integer>
## concurrent number of threads getting Cloud Cover PSA value from xml file
CONCURRENT_GET_CLOUDCOVERP=<integer>
DEBUG_MESSAGES=<string Y/N>

```

The table 4.8.21-1 describes the individual configuration parameters mentioned above:

Table 4.8.21-1. Data Pool Access Configuration Parameters for Rollup Scripts

Parameter Name	Description
PGM_ID	Used to decrypt the database connection password for the database user represented by this utility.
SYBASE_HOST	The Sybase database host to connect to.
SYBASE_PORT	The Sybase database port on the specified host.
SYBASE_DPL_DBNAME	The name of the Data Pool database to connect to.
SYBASE_USER	The user name for the Sybase connection.
SYBASE_JDBC_DRIVER_CLASS	The name of the SQL server for this Sybase connection.
DB_RETRIES	The number of times the utility attempts to connect to the database. The recommended default is 5.
DB_SLEEPSECONDS	The number of seconds the utility waits ('sleeps') between connection attempts. The recommended default is 15.
DB_BATCH_SIZE	The batch size for processing the database requests. The recommend default is 2000.
SQL_TIMEOUT_SECONDS	The number of seconds after which a pending / hung SQL command will be time out. The recommend default is 3600.
CONCURRENT_GET_CLOUDCOVERP	The number of concurrent threads used when executing operations (populate or repopulate) of the cloud cover collections for which the cloud cover source is a PSA. Concurrency is needed for performance reasons because the cloud cover value must be read from the granule XML file. Please note that the utility will use CONCURRENT_GET_CLOUDCOVERP + 1 database connections while it runs.
DEBUG_MESSAGES	Y – provides verbosity in the debug log, N – provides basic messages in the debug log.

4.8.21.3 Databases

The Data Pool Cloud Cover Utility script uses the Sybase ASE Server.

4.8.21.4 Special Constraints/Dependencies

The Data Pool Cloud Cover Utility script functions only if the Data Pool database is available. For operations that require parsing of XML files (when the cloud cover source is a PSA), access to the file system where the collection XML files are stored is required.

4.8.21.5 Outputs

The Data Pool Cloud Cover Utility script records its actions in its log file EcDICloudCoverUtility.log.

4.8.21.6 Event and Error Messages

All event and error messages generated from the Data Pool Cloud Cover Utility script are written to the log file EcDICloudCoverUtility.log. DICcuRequests table records the status of all the Cloud Cover Utility runs performed.

4.8.21.7 Reports

None

4.8.21.8 Recovery Procedures

In the case that the script crashes or is killed during a run, the script can be re-run with the same options.

Glossary

AutoSys/AutoXpert	COTS software that provides job scheduling and management. Also provides graphics to monitor, analyze, forecast and plan AutoSys implementations.
Baseline Manager	Baseline Manager package used to maintain records of baselined operational system configurations. (see also XRP-II)
Batch Insert Utility	The Batch Insert Utility is a command line interface that allows operators to insert granules residing in or outside of (non-ECS granules) the ECS archive into the Data Pool.
Bulk Metadata Generation Tool	The EcOsBulkURL Utility allows operators to make available the File Transfer Protocol (FTP) Universal Resource Locators (URLs) in the Data Pool to the ECS Clearing House (ECHO).
ClearCase	Software change manager that stores ECS custom software and science software, regulates access to the files, controls and logs file changes, performs software builds, and maintains a record of the build. Maintains a library of software deployed to sites.
CMI	Cryptographic Management Interface. Used to create accounts for given user names and passwords.
Crack	Used to determine if passwords are secure.
DAR	Data Acquisition Request for ASTER instrument data.
Database Installation and Maintenance Scripts	A set of eleven standard database scripts have been created for the DDIST, INGEST, MSS, STMGT, and SUBSRV subsystems to facilitate database installation and database administration activities. These scripts are designed to be accessible from both the command line and the Stage Install function of ECSAssist.
Data Dictionary Maintenance Tool	Tool that allows the operator to maintain the ECS Data Dictionary.
Data Distribution Requests GUI	Monitors and controls the request for data distribution (for FTP Pushes and FTP Pulls only).
Data Ingest	Provides a means for external providers to ask for ECS ingest services.

Data Pool Ingest GUI	The Data Pool Ingest Graphical User Interface allows the operators to view past ingest activities, monitor and control ingest requests, modify system and external data provider parameters, and initiate hard media ingest.
Data Pool Access Statistics Utility – Rollup Scripts	The Data Pool Access Statistics Utility (DPASU) provides the ECS Operations Staff with several capabilities related to collecting access statistics for the Data Pool database. The DPASU encompasses two types of scripts: rollup and maintenance. The rollup scripts read and parse access logs to compile statistics and store those records in the Data Pool database, while the maintenance scripts backup, restore, and delete data in the related Data Pool database tables.
Data Pool Access Statistics Utility – Maintenance Scripts	The Data Pool Access Statistics Utility (DPASU) provides the ECS Operations Staff with several capabilities related to collecting access statistics for the Data Pool database. The DPASU encompasses two types of scripts: rollup and maintenance. The maintenance scripts backup, restore, and delete data in the related Data Pool database tables.
Data Pool Cleanup Utility	The Data Pool Cleanup utility provides a mechanism for the ECS Operations Staff to remove expired granules and their associated metadata and browse files from the Data Pool disks and corresponding Data Pool database inventory.
Data Pool Maintenance GUI	The DPM GUI provides an operator interface to monitor the current status of Data Pool Inserts and to maintain specific Data Pool parameters. This GUI manages ECS and Non-ECS data collections.

Data Products	<p>Designated as standard or special data products, generated as a part of research investigation using EOS data. The various levels of data are defined as follows (1995 MTPE/EOS Reference Handbook):</p> <p><i>Level 0</i> - Reconstructed, unprocessed instrument/payload data at full resolution; any and all communications artifacts, e.g., synchronization frames, communications headers, duplicate data removed.</p> <p><i>Level 1A</i> - Reconstructed, unprocessed instrument data at full resolution, time-referenced, and annotated with ancillary information, including radiometric and geometric calibration coefficients and geo-referencing parameters, e.g., platform ephemeris, computed and appended but not applied to the Level 0 data.</p> <p><i>Level 1B</i> - Level 1A data that have been processed to sensor units (not all instruments will have a Level 1B equivalent).</p> <p><i>Level 2</i> - Derived geophysical variables at the same resolution and location as the Level 1 source data.</p> <p><i>Level 3</i> - Variables mapped on uniform space-time grid scales, usually with some completeness and consistency.</p> <p><i>Level 4</i> - Model output or results from analyses of lower level data (e.g., variables derived from multiple measurements).</p>
Data Server	<p>Software associated with storing earth science and related data, searching and retrieving the data, and supporting the administration of the data, hardware devices, and software products.</p>
DDTS	<p>Change request manager used to compose, submit, report and track status of proposals to change ECS resources electronically.</p>
ECS Assistant	<p>The ECS Assistant (ECSAssist) is a custom program that simplifies the process of installation, testing and management of ECS. The tool is for use in installing software and maintaining the information related to that software. Only the Subsystem Manager function of ECSAssist should be used in the ECS operational environment.</p>
ECS Desktop	<p>Simulates Common Desktop Environment (CDE); interface that acts like a file manager, allowing launch of applications, creation of directories and moving/copying/ deleting files.</p>

ECS Registry GUI	The ECS Registry GUI is a management tool for ECS applications allowing users to create and update parameter information. Registry data is stored in a registry database.
Email	Service that manages electronic mail messages for DAAC operators.
EOSView	A custom HDF file verification tool. Displays HDF files and HDF-EOS data.
Event Log	The Event Log Database resides at each ECS site. It records status and error messages generated by the various ECS applications at the site. The Event Log Browser is used to view the status and error messages.
Event Log Directory	This directory resides on every computer platform and contains the log files used by applications to report status and error messages. Log files in the Event Log Directory are loaded into the Event Log Database on a periodic basis.
FLEXIm	COTS for the administration of licenses.
GCMD Data Export	Extracts Data Interchange Format (DIF) from the AIM inventory database to the Global Change Master Directory (GCMD).
Granule Deletion Administration Tool	The Granule Deletion Administration Tool provides the ECS Operations Staff with the capability to delete granules using a command line interface. The granules can be deleted from both the inventory and archive or just the archive. Granules are not physically deleted from the archive. The directory entry is deleted so the files cannot be accessed. The physical storage occupied by the deleted granules is not reclaimed through this operation.
Hyperic	A computer system and network monitoring application software that provides the ability to discover, organize, and monitor resources.
IDL	Interactive data language used to interactively visualize and analyze scientific and engineering data products.
Ingest GUIs	Allows monitor and control of Ingest requests, modification of system and external data-provided parameters and initiate hard media ingest. An HTML interface allows for submission of ingest requests for processing.
Inventory, Logistics and Maintenance (ILM) Manager	Supports M&O in maintaining records that describe all inventory components, structures, and interdependencies.

ISQL	SQL command parser utility used to interact with a SQL server and databases on a SQL server.
Java System Web Server	This COTS product is a multi-process, multi-threaded, secure web server built on open standards. It provides high performance, reliability, scalability, and manageability for any size enterprise, and it includes modules for creating and managing Web content, for extending or replacing functions of the server (e.g., through Java servlets and JavaServer pages), and for providing application-specific services such as security and access control. In ECS, the Web Server is used by several subsystems to access HTML files and to service web-based applications.
Main Window Manager	Provides login to UNIX and ECS, authenticates the user and brings up the appropriate ECS Desktop based upon the operator role.
Microsoft Office Professional	A collection of applications working together as if it were a single program. The collection includes Word (for text and graphical processing), Excel (a spreadsheet) and PowerPoint (making graphics/presentations) programs.
Netscape Communicator	World Wide Web (WWW) browser. Used to obtain information from other sources.
Networker	Tool used by system administrators to perform site-wide system backups, except databases.
Order Manager GUI	The Order Manager (OM) Graphical User Interface (GUI) provides the operators with direct access to the OM database. The GUI allows operators to view and modify requests that have been placed on hold by the Order Manager because they require operator intervention and resubmit requests or portions of a request that failed. For Synergy III, the GUI is an addition to the existing System Management Subsystem (MSS) Order Tracking GUI and the Data Distribution (DDIST) GUI rather than a replacement for them.
Order Manager Command Line Utility	The Order Manager Command Line utility provides a mechanism by which the ECS Operations Staff can submit order requests into the Order Manager System (OMS) database directly without knowing whether the Order Manager Server is up or down.
Order Tracking	User services tool that tracks order status and request status.

PIPRGenerator User Interface	The PIPRGenerator is the command line interface for the Production Request Editor. The PIPRGenerator allows the user to create and activate a number of Routine Production Requests using information contained in an input file. The input file contains the PgeIds and GEOIds for the PGEs and primary input granules, respectively, for the Production Requests to be created.
Process Control File	Specifies the names and locations of files used by science software executables, and defines the correspondence between the file specifications and the logical identifiers used by the science software to reference the specified files.
Quality Assurance Monitor	The ECS Quality Assurance (QA) Monitor processing capabilities enable DAAC operators to perform duties associated with DAAC QA activities. The ECS QA Monitor GUI is the user-interface for entering data requests and displaying data, status, and error messages. The QA Monitor does not produce data products, but communicates with the science data server to retrieve data that have been previously archived. The ECS QA Monitor assists in manual quality assurance activities such as querying and retrieving data granules, visualizing data products and updating metadata.
Regenerate Failed PDR Tool	The Regenerate Failed PDR tool provides the ECS Operations Staff with the capability to generate a Product Delivery Record (PDR) for each failed granule in a PDR and copy the generated PDRs to an Ingest polling directory using a command line interface. The purpose of the tool is to provide a means for the ECS Operations Staff to easily resubmit only failed granules to Ingest polling, rather than having to manually edit the original PDR file or resubmit all of the granules, which would create duplicate granules in the archive.
Replication Server	Maintains warm standby copies of application data and replicates changes among databases at different sites.
Resource Planning	Used to plan the allocation of DAAC resources.

Restricting ESDT and Granule Access	The two scripts <i>EcDsSrUpdateESDTAccess</i> and <i>EcDsSrUpdateQATimeRange</i> provide DAAC operations staff the capability to adjust how the Science Data Server restricts <i>Acquire</i> access to granules. When evaluating a user's permission to <i>Acquire</i> a granule, the Science Data Server uses the value of the NASA user attribute stored in the User Profile system. The first script, <i>EcDsSrUpdateESDTAccess</i> , allows the DAAC operator to restrict an entire ESDT/Data Collection to one or more of the specific NASA user types stored in the User Profile system. The second script, <i>EcDsSrUpdateQATimeRange</i> , allows individual granules to be restricted based upon the granule's QA flag values and the type of NASA user making the request.
Sniffers	Monitors network traffic for collisions and troubleshooting.
SQL Server	A SQL Server is a set of cooperating processes that manage multiple Sybase databases and multiple users.
SSI&T Manager	Allows check in and verification of science software delivered by the instrument teams at the Science Computing Facilities. Provides access to all COTS tools and custom applications that are part of the SSI&T environment.
StorNext	StorNext Storage Manager (SNSM) is a hierarchical storage management (HSM) system for managing data on multiple storage tiers consisting of disk and tape resources.
Subscription Editor	Allows the operator to manually enter Subscriptions to the Subscription Server.
Subscription Server	Allows users to register their events related to a certain type of data.
Sybase Replication Server	Maintains warm standby copies of application data and replicates changes among databases at different sites.
TCP Wrappers	Monitors and controls access to network services on a host.
TestTrack Pro	TestTrack Pro (TTPro) provides a trouble ticketing service that furnishes both ECS users and operations personnel at the DAACs a common environment for classifying, tracking, and reporting the occurrence and resolution of system-related problems.
Tripwire	An intrusion detection tool that monitors files for changes.

Tuple	Data reflecting unique strings of information associated with and descriptive of an event (e.g., names, identifier numbers, data types).
Update Granule	The Update Granule Utility provides the ECS Operations Staff with a command line interface to update the expiration date and optionally the retention priority of granules in the Data Pool inventory. The granules in the Data Pool inventory can be ECS or non-ECS granules.
User Account Management GUI	Tool used by DAAC operators to process new accounts and manage existing ones.

Abbreviations and Acronyms

A

ACS	Automated Cartridge System
ACSLs	Automated Console System for Library Services
ADC	Affiliated Data Center
AI&T	Algorithm Integration and Test
AITTL	Algorithm Integration and Test CSCI
ALOG	Application Log file
AM-1	See Terra
AMASS	Archival Management and Storage System
AML	Automated Media Library
AMU	Automated Management Unit
ANSI	American National Standards Institute
AOI	Area of Interest
AOS	Area of Search
API	Application Program (or programming) Interface
AR	Action Request
ASBP	AIRS Summary Browse Products
ASCII	American Standard Code for Information Exchange
ASTER	Advanced Spaceborne Thermal Emission and Reflection Radiometer

B

BB	Bulletin Boards
BIS	Baseline Information System
BLM	Baseline Manager
BOM	Bill of Material
BMGT	Bulk Metadata Generation Tool

C

CAP	Cartridge Access Port
CCB	Configuration Control Board
CCR	Configuration Change Request
CCS	Control Center System Middleware, a custom code replacement for DCE
CD	Cartridge Drives
CD-ROM	Compact Disk -- Read Only Memory
CDDTS	Clear Distributed Defect Tracking System
CDE	Common Desktop Environment
CDRL	Contract Data Requirements List
CERES	Clouds and the Earth's Radiant Energy System
CFG	Configuration File
CGI	Common Gateway Interface
CHUI	Character-based User Interface
CI	Configuration Item
CID	Control Item Identifier
CIDM	Client, Interoperability and Data Management Subsystem group
CLI	Command Line Interface
CLS	Client Subsystem
CM	Configuration Management
CMI	Cryptographic Management Interface
COTS	Commercial Off-The-Shelf
CPIO	Copy In and Out
CPU	Central Processing Unit
CRM	Change Request Manager
CSCI	Computer Software Configuration Item
CSDT	Computer Science Data Type
CSMS	Communications and Systems Management Segment
CSS	Communications Subsystem (of CSMS)
CSV	Comma Separated Variable

D

DAAC	Distributed Active Archive Center
DAO	Data Assimilation Office (at GSFC)
DAP	Delivery Archive Package Delivery Algorithm Package
DAR	Data Acquisition Request
DAS	Data Availability Schedule Distributed Archive Server
DB	DataBase
DBA	Database Administrator
DBMS	DataBase Management System
DBO	Database Owner
DCCI	Distributed Computing Software CSCI (of CSS)
DDICT	Data Dictionary CSCI (of DMS)
DDMT	Data Dictionary Maintenance Tool
DDTS	Distributed Defect Tracking System (COTS)
DEG	Degrees
DES	Data Encryption Standard
DHWM	Data High Water Mark
DID	Data Item Description
DIF	Data Interchange Format
DLL	Dynamically Linked Library Data Link Library
DLT	Digital Linear Tape
DLWM	Data Low Water Mark
DMS	Data Management Subsystem (of SDPS) Degrees, Minutes and Seconds
DNS	Domain Name Service
DO	Derived Objects
DPAD	Data Pool Action Driver

DPASU	Data Pool Access Statistics Utility
DPIU	Data Pool Insert Utility
DPL	Data Pool
DPM	Data Pool Maintenance
DPR	Data Processing Request
DSKT	Desktop CSCI (of CLS)
DSS	Data Server Subsystem (of SDPS)
DTS	Distributed Time Service

E

EA	External Ancillary
EASI	ECS Assist System Installation
EBIS	ECS Baseline Information System
ECHO	ECS Clearing House
ECN	Equipment Control Number
ECS	EOSDIS Core System
EDOS	EOS Data and Operations Ssystem
EDC	EROS Data Center (DAAC)
EDF	ECS Development Facility
EDGRS	ESDIS Data Gathering and Reporting System
EDHS	ECS Data Handling System
EEB	EMD to EED Bridge
EIF	Entry Interface Facility
EIN	Equipment Inventory Number
EOC	EOS Operations Center (ECS)
EOS	Earth Observing System
EOS-AM	EOS Morning Crossing (Descending) Mission -- see Terra
EOSDIS	Earth Observing System Data and Information System
ESDIS	Earth Science Data and Information System
ESDT	Earth Science Data Type

ESN EOSDIS Science Network
ESSM Enterprise SQL Server Manager
ETAC EMASS Technical Assistance Center

F

FDDI Fiber Distributed Data Interface
FIFO First In-First Out
FQDN Fully Qualified Domain Name
FSMS File Storage Management System
FTP File Transfer Protocol

G

GB Giga-Byte
GCMD Global Change Master Directory (not developed by the ECS project)
GFE Government Furnished Equipment
GMT Greenwich Mean Time
GSFC Goddard Space Flight Center (DAAC)
GTWAY Gateway CSCI (of DMS)
GUI Graphical User Interface
GV Ground Validation

H

HAIF HDF ASCII Interchange Format
HDF Hierarchical Data Format
HDF-EOS an EOS proposed standard for a specialized HDF data format
HEG HDF-EOS to GeoTIF Converter
HMI Human Machine Interface
HQ Hyperic HQ Enterprise
HQU HQ User Interface
HTML HyperText Markup Language

HTTP Hypertext Transport Protocol
HWCI Hardware Configuration Item

I

I&T Integration and Test
I/O Input/Output
ICD Interface Control Document
ICMP Internet Control Message Protocol
ID IDentification
IDG Infrastructure Development Group
IDL Interactive Data Language
ILM Inventory, Logistics and Maintenance Manager
IMSL International Math and Statistics Library
INCI Internetworking CSCI (of CSMS)
INGST Ingest Services CSCI (of INS)
INS Ingest Subsystem (of SDPS)
IOS Interoperability Subsystem
IP Internet Protocol (address)
IRD Interface Requirements Document
ISO International Standards Organization
ISQL Interactive Structured Query Language
ISS Internetworking Subsystem (of CSMS)
IT Instrument Team

J

JPL Jet Propulsion Laboratory

L

L0-L4 Level 0 (zero) through Level 4 data
LAN Local Area Network

LaRC	Langley Research Center (DAAC)
LCU	Library Control Unit
LDAP	Lightweight Directory Access Protocol
LMU	Library Management Unit
LSM	Library Storage Module
LTM	Log Transfer Manager

M

MB	MegaByte (10^6 bytes)
MCF	Metadata Configuration File
MCI	Management Software CSCI
MDA	Management Data Access
MFR	Manufacturer
MI	Machines Impacted
MIN	Minutes
MISR	Multiangle Imaging SpectroRadiometer
MLCI	Management Logistics CSCI
MM	Millimeter
MODIS	Moderate-Resolution Imaging Spectrometer
MOPITT	Measurements of Pollution in the Troposphere
MSS	System Management Subsystem (of CSMS)
MTPE	Mission to Planet Earth
MUA	Mail User Agent
MWO	Maintenance Work Order

N

N/A	Not Applicable
NBSRV	Spatial Subscription Server
NCR	Non-Conformance Report

NCS	Network Computing System
	Netscape Commerce Server
NESDIS	National Environmental Satellite Data and Information Service
NFS	Network File System
NMC	National Meteorological Center (NOAA)
NOAA	National Oceanic and Atmospheric Administration
NSBRV	Spatial Subscription Server CSCI
NSIDC	National Snow and Ice Data Center (DAAC)
NW	NetWorker

O

ODFRM	On-Demand Product Request Form (of CLS)
ODL	Object Description Language
OEM	Original Equipment Manufacturer
OODCE	Object Oriented DCE
OM	Order Manager
OMS	Order Manager Server
OPER	Operator
OPS	Operations
OS	Operating System
OSF	Open Systems Foundation
OSI	Open System Interconnect
OTS	Off-the-Shelf

P

PAN	Production Acceptance Notification
PCF	Process Control File
PCFG	Parameter Configuration File
PDF	Portable Document Format
PDR	Production Data Request

	Product Delivery Record
PFC	Prohibited Function Checker
PGE	Product Generation Executable
PLANG	Production Planning CSCI (of PLS)
PM	Preventative Maintenance
PO	Purchase Order
POSIX	Portable Operating System Interface for Computer Environments
PR	Production Request
PRS	Primary Replication Server
PRONG	Data Processing CSCI (of DPS)
PVC	Performance Verification Center
PWB	Planning Work Bench (of PLS)
	Production Planning Workbench

Q

QA	Quality Assurance
QC	Quality Control
QRU	Query/Retrieve/Update

R

RAID	Redundant Array of Inexpensive Disks
RAM	Random Access Memory
RMA	Reliability, Maintainability, Availability
ROC	Read-Only Cache
RRS	Replicate Replication Server
RSA	Replication System Administration
RSI	Replication Server Interfaces
RSM	Replication Server Manager
RSSD	Replication Server System Database
RTF	Rich Text Format

RTU Rights To Use

S

SA System Administrator

SAA Satellite Active Archive

SAGE Stratospheric Aerosol and Gas Experiment

SCF Science Computing Facility

SCLI Science Data Server Command Line Interface

SCSI Small Computer System Interface

SDP Science Data Processing

SDPF Sensor Data Processing Facility (GSFC)

SDPS Science Data Processing Segment (ECS)

SDPTK SDP Toolkit CSCI

SDS Science Data Standards (Science Data Group data used in EOSView)

SDSRV Science Data Server CSCI (of DSS)

SEC Seconds

SGI Silicon Graphics, Inc.

SMC System Management Center (ECS – at GSFC)

SMTP Simple Mail Transport Protocol

SNMP Simple Network Management Protocol

SP Space Pool

SPRHW Science Processing HWCI

SQL Structured Query Language

SQR SQL Report Writer

SSH Secure Shell

SSI&T Science Software Integration and Test

SSM/I Special Sensor for Microwave/Imaging (DMSP)

SSO System Security Officer

SSS Spatial Subscription Server

SST Sea Surface Temperature
SSAP Science Software Archive Package
STK Storage Tek
SYS System

T

TAR Tape Archive
TCP/IP Transmission Control Protocol/Internet Protocol
TDP Tabular Datastream Protocol
Terra EOS AM Project spacecraft 1, morning spacecraft series -- ASTER, MISR, MODIS and MOPITT instruments (formerly called AM-1 spacecraft)
TOMS Total Ozone Mapping Spectrometer
TONS TDRS On-board Navigational System
TT Trouble Ticket
TTPro TestTrack Pro

U

UFS UNIX File System
UR Universal Reference
URL Universal Resource Locator
USGS United States Geological Survey
UT Universal Time
UTC Universal Time Code
UUID Universal Unique Identifier

V

VATC Verification and Test Center
VOB Version Object Base

W

WAIS Wide Area Information Server

WAN Wide Area Network

WKBCH Workbench CSCI (of CLS)

X

xAR (generic) Acquisition Request

XML Extensible Markup Language