4.8 Common Services Tools

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

4.8.1 Red Hat Enterprise Linux GNOME Desktop Environment
4.8.2 Firefox
4.8.3 Oracle iPlanet Web Server
4.8.4 Data Pool Access Statistics Utility (DPASU) – Rollup Scripts
4.8.5 Data Pool Access Statistics Utility (DPASU) – Maintenance Scripts
4.8.6 Most Recent Data Pool Inserts Utility
4.8.7 Data Pool Collection-to-Group Remapping Utility
4.8.8 QA Update Utility (QAUU)
4.8.9 Data Pool Move Collections Utility
4.8.10 Data Pool Hidden Scrambler Utility
4.8.11 Data Pool Band Backfill Utility
4.8.12 XML Replacement Utility
4.8.13 DPL Cleanup Orphan/Phantom Validation Utility (EcDlCleanupFilesOnDisk .pl)
4.8.14 DataPool Cleanup Granules Utility (EcDlCleanupGranules.pl)
4.8.15 Link Checker Utility (EcDlLinkCheck.pl)
4.8.16 AIM Tape Archive Consistency Checking Utility (EcDsAmArchiveCheckUtility)
4.8.17 XML Archive Metadata Check Utility EcDsCheckXMLArchive.pl
4.8.18 Cloud Cover Utility (EcDlCloudCoverUtilityStart) (Obsolete in Release 8.2)
4.8.19 DPL XML Check Utility
4.8.20 AIM Map Generation Server (MGS)
4.8.21 AIM Map Generation Utility (MGU)
4.8.22 EOSDIS Service Interface (DataAccess)
4.8.23 DataPool Checksum Verification Server
4.8.24 OM CleanUp Old Data Utility
4.8.25 OM DB CleanUp Archive Utility

This page intentionally left blank.

4.8.1 Red Hat Enterprise Linux GNOME Desktop Environment

The ECS uses the Red Hat Enterprise Linux GNOME Desktop Environment to manage X windows. It is a graphical user interface for Linux. It provides users registered at an ECS site with generalized support for performing the basic operations. Some of the operations are listed in Table 4.8.1-1.

Table 4.8.1-1. ECS Operator Functions Performed using Linux GNOME Desktop

Operating Function	GUI	Description	When and Why to Use
Start a desktop session	Basic login with userid and password	Invokes the Linux GNOME desktop.	Access an ECS host.
Use the Front Panel	Front Panel window	Contains set of controls for performing common tasks (i.e., calculator, text editor, graphics, system preferences, 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	Configuration Editor - schemas	Allow for customizing the look and behavior of desktop.	Need to customize desktop environment.
Use text editor	Text Editor	Supports creation/editting 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	Opens a command line terminal window on the desktop.	Need to access command line terminal window.

4.8.1.1 Quick Start Using the Linux GNOME Desktop Environment

After being registered as an ECS user by the site administrator, the user accesses the Linux GNOME Desktop by logging into an ECS host using a defined UserID and password.

4.8.1.2 Main Screen

Figure 4.8.1-1 presents an example of the type of support provided by the Linux GNOME Desktop.

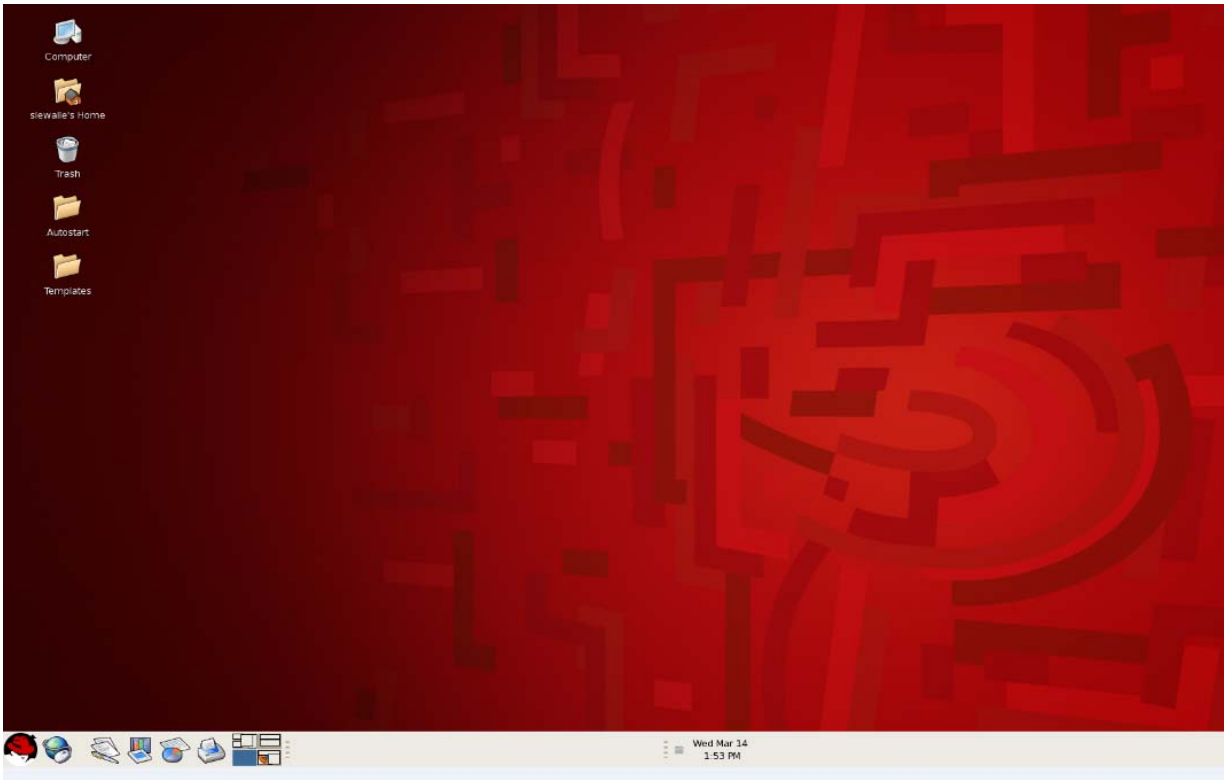


Figure 4.8.1-1. Example of Linux Desktop Features

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 open an internet browser, see date and time, select the Print Manager, view multiple desktop windows, and access the Red Hat menus.

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

The Terminal window provides Linux command line access to operating system services.

4.8.1.3 Required Operating Environment

Red Hat Enterprise Linux.

4.8.1.4 Databases

None

4.8.1.5 Special Constraints

Access to Red Hat Enterprise Linux GNOME Desktop is available only to registered users of ECS sites.

4.8.1.6 Outputs

The Linux GNOME Desktop only outputs event and error messages.

4.8.1.7 Event and Error Messages

The Linux GNOME Desktop issues both status and error messages to the operator screen. Error messages are listed in the support documentation accessible with a user login at the web link:

http://redhat.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	• 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	• 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:

> **/usr/bin/firefox &**
or
> /usr/ecs/OPS/COTS/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.

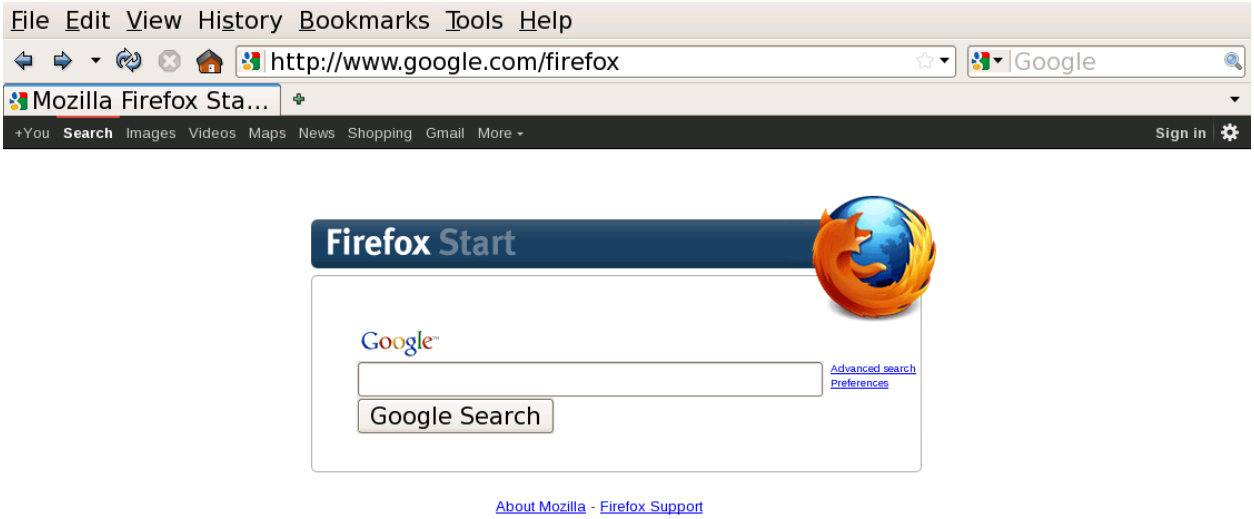


Figure 4.8.2-1. Firefox: Browser with Default Home Page

Operators can configure Firefox to use a different Home Page as the default. In addition, individuals can set a preferred Home Page by clicking **Edit → Preferences** on the menu bar, entering a URL in the Home Page field, then clicking **Close**. 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 Oracle iPlanet Web Server

Oracle iPlanet 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 the ECS, Oracle iPlanet 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 ECS applications relying on it. A distinct instance of a Oracle iPlanet Web Server is created for each such application, one per mode in which the application runs. For example, ECS Order Manager, Data Pool GUI, and Spatial Subscription Server all need to use Oracle iPlanet Web Server, and each of them runs in the three modes on sites' Internal Interface Server machines. Consequently, nine instances of the Oracle iPlanet 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 iPlanet Web Servers use can be found in the ECS baseline document, 910-TDA-002, ECS Software Port Mapping Baseline.

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

Table 4.8.3-1 summarizes the Oracle iPlanet Web Server functions used by ECS and references vendor guides that describe their use. Release Notes and other iPlanet documents are available on the Internet at http://docs.oracle.com/cd/E18958_01/index.htm.

Table 4.8.3-1. Common ECS Operator Functions Performed with the Oracle iPlanet Web Server (1 of 3)

Operating Function	Command/Script	Description	When and Why to Use
Administer *Oracle iPlanet* 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: • 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 *Oracle iPlanet* is installed and when the Administration Server needs reconfiguration.
Provide security and encrypt transactions	Configurations GUI	Allow operators to: • 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 ECS Operator Functions Performed with the Oracle iPlanet Web Server (2 of 3)

Operating Function	Command/Script	Description	When and Why to Use
Configure web servers	Configurations GUI	Allows operators to: • 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 ECS custom code.
Analyze log files	Configurations GUI Virtual Servers GUI	Allows operators to: • 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: • 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 ECS Operator Functions Performed with the Oracle iPlanet Web Server (3 of 3)

Operating Function	Command/Script	Description	When and Why to Use
Program the server	Virtual Servers GUI	Allows operators to: • 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: • 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 Oracle iPlanet Web Server

Oracle iPlanet 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 *Oracle iPlanet Web Server 7.0.9 CLI Reference Manual* for a detailed description of wadm shell commands, and to 914-TDA-541 Rev02, *Task 2 Rev 18 Linux COTS Software Maintenance Upgrade for Release 8.2 for the EOSDIS Core SYSTEM (ECS)* for examples of how wadm can be used to create web server instances needed by ECS.

Note: ECS currently uses multiple instances of the web server rather than multiple virtual servers in a web server configuration. In past releases, Oracle iPlanet 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-<*xxxxxxx*> start

where <*xxxxxxx*> 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 3.6, included in the ECS baseline, is capable of supporting both frames and Java Script.

4.8.3.2 Oracle iPlanet Web Server Main Screen

The first page you see when you access the Administration Console is the Common Tasks page. This is Oracle iPlanet'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.

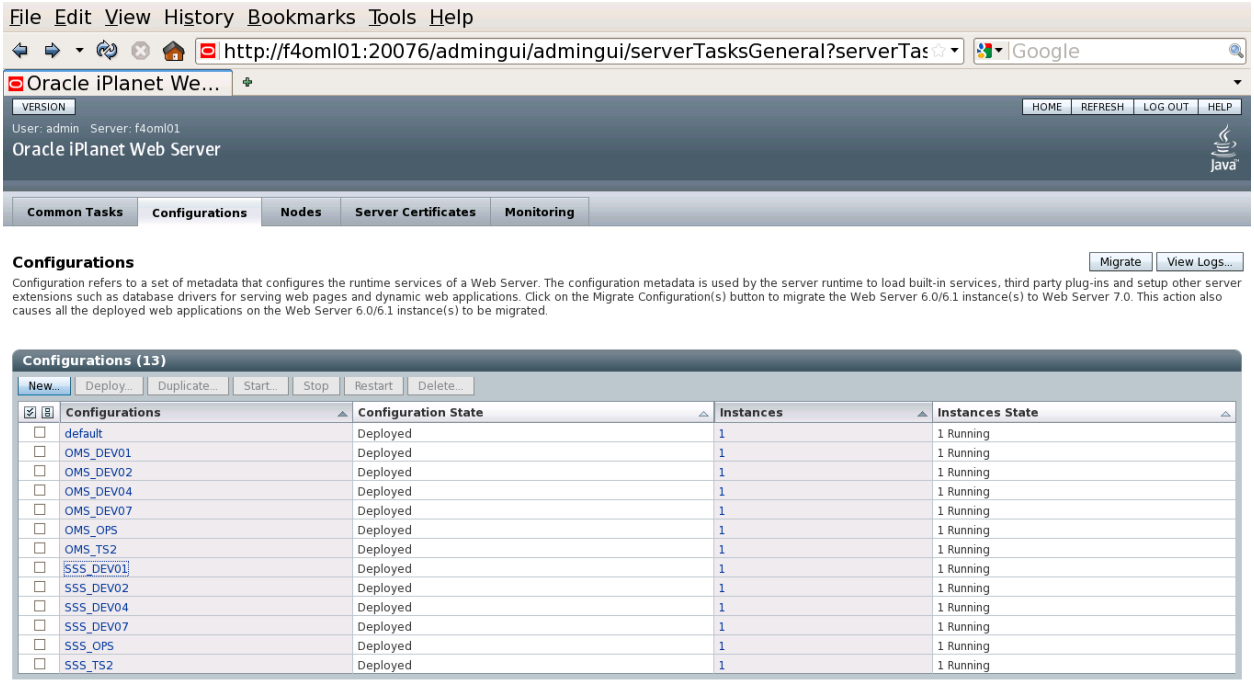


Figure 4.8.3-1. Oracle iPlanet Web Server: Common Tasks Screen

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 pull-down 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 Oracle iPlanet 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 Oracle iPlanet 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 Oracle iPlanet Web Server: Configurations screen.

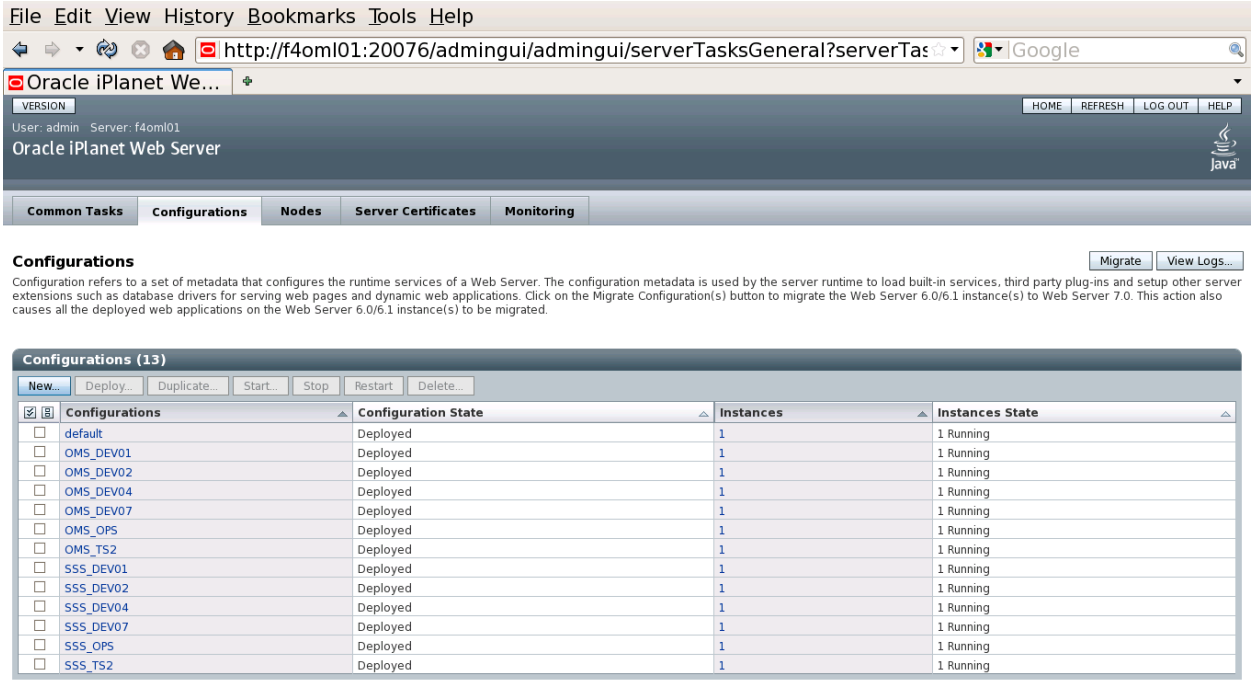


Figure 4.8.3-2. Oracle iPlanet Web Server: Configurations Screen

4.8.3.2.2 Oracle iPlanet 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 Oracle iPlanet Web Server: Virtual Servers Screen.

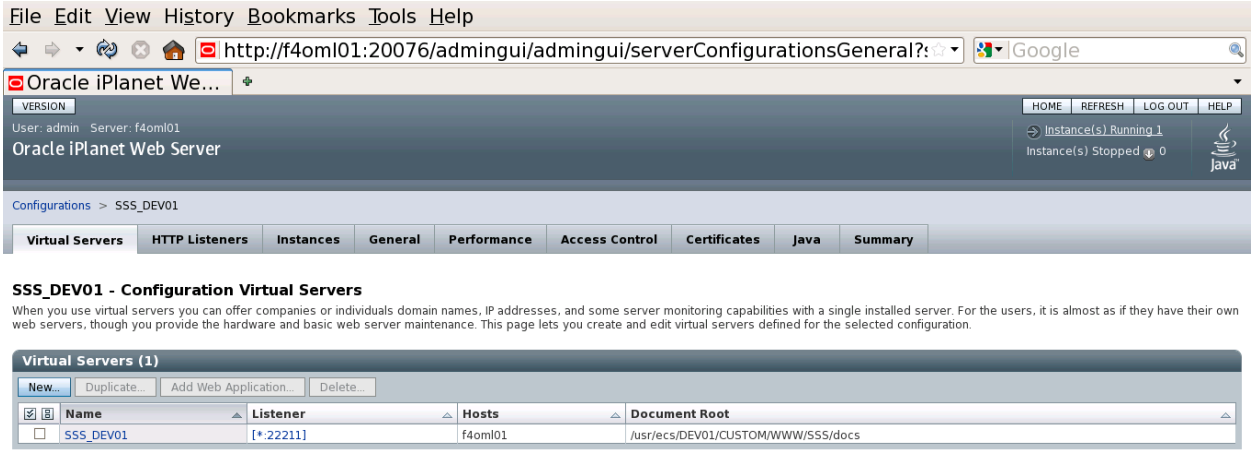


Figure 4.8.3-3. Oracle iPlanet Web Server: Virtual Servers Screen

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 Oracle iPlanet 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 3.6.24 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 Oracle iPlanet 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, Oracle iPlanet 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 ECS does not currently use them. Refer to the *Oracle iPlanet Web Server 7.0.9 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 Oracle iPlanet'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 Oracle iPlanet Web Server 7.0.9 *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 *Oracle iPlanet Web Server 7.0.9 Administrator's Guide* and the *Oracle iPlanet Web Server 7.0.9 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

This page intentionally left blank.

4.8.4 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.

4.8.4.1 Data Pool Access Rollup Scripts

The Data Pool access rollup scripts provide the ECS Operations Staff with the capability to parse the EOSDIS Service Interface (ESI) Data Access and FTP/HTTP 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/HTTP access logs, the rollup scripts store into the Data Pool database (in the DlGranuleAccess 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 accesses through ESI Data Access (EcDlDaRollupApacheLogs.ksh), one to track access to the Data Pool via anonymous ftp (EcDlRollupWuFtpLogs.pl) and a third to track access to the Data Pool via httpd (EcDlRollupHttpLogs.pl). All three rollup scripts are installed and run on the externally facing host, x4eil01.

The EcDlDaRollupApacheLogs.ksh script loads a file generated by the Apache logger in the mode's data/DPL directory into the AmDaTransfer table. The EcDlRollupWuFtpLogs.pl script parses the wu-ftp log written to /var/log/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 4.8.4.3.3 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. EcDlDaRollupApacheLogs.ksh script does not need any of these options; it takes care of loading all the current data into the AmDaTransfer table and renames the file containing that data by appending a time stamp. Care must be taken not to remove the file located at f4eil01:/usr/ecs/<MODE>/CUSTOM/data/DPL/ EcDlDataAccessApache.log.

The other Data Pool access rollup scripts work as follows. The script first parses the specified FTP or HTTP 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 (DlFtpAccessLog). 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 DlGranuleAccess table. The flat file is removed and an entry is made into the DlAccessRollup 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 DlAccessRollup 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.

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

Entering the following commands run the rollup scripts:

% **EcDlDaRollupApacheLogs.ksh <no options necessary>**

% **EcDlRollupWuFtpLogs.pl <command line parameters>**

% **EcDlRollupWuHttpLogs.pl <command line parameters>**

There are various optional and required command line parameters used in combination with each other. Table 4.8.4-1 provides a description of these parameters.

Table 4.8.4-1. 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.
-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.

4.8.4.1.2 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 4.8.4.3.3 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.

4.8.4.1.3 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.

4.8.4.1.4 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 DlGranuleAccess table.)

d. The rotation/renaming times of the Web Access and 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 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 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 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 by either 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 command line parameter, but do NOT need to be enclosed in quotes if used with the configuration parameter WEB_LOG_PATH. See sections 4.8.5.1.5 and 4.8.5.1.6).

e. To prevent or minimize the chances of database contention, it is recommended the daily cron job for rolling up 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.

4.8.4.1.6 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.

4.8.4.1.7 Command-line Examples

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

<u>Example 1:</u>

EcDlRollupWebLogs.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.

<u>Example 2:</u>

EcDlDaRollupApacheLogs.ksh Run ESI Data Access rollup script in. This example is typical of syntax used in the crontab file.

<u>Example 3</u>

EcDlRollupFwFtpLogs.pl OPS –start 2002/02/15 –fwftp "/usr/logs/.log "*

EcDlRollupWuFtpLogs.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.

4.8.4.2 Data Pool Access Statistics Main Screen

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

4.8.4.3 Required Operating Environment

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

4.8.4.3.1 Interfaces to supporting products

Table 4.8.5-2 lists the supporting products that this tool depends upon in order to function properly.

Table 4.8.4-2. Interface Protocols

Product Dependency	Protocols Used	Comments
AIM database	SQL	Via SQL server machines
Perl DBI for postgres	DBI	Requires proper install of Perl

4.8.4.3.2 Access Log File Formats

The Data Pool access rollup scripts are dependent on a particular format of both the 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 FTP/HTTP or Apache access log formats change in any way. Subsequent sections provide format details for these access logs.

4.8.4.3.3 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 ***EcDlWebRollup.CFG*** contains the configuration parameters for the Web Rollup script. The file EcDlWuFtpRollup.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:

```
DBUSERNAME = EcDlFtpRollupStats
DBSERVER = x4dbl03
DBNAME = ecs
DBSUBSYSTEM=aim
NUM_RETRIES = <integer>
SLEEP_SEC = <integer>
FTP_LOG_PATH = <path and file name>
ROLLUP_START_TIME = <time of day>
```

Table 4.8.4-3 describes the individual configuration parameters mentioned above.

Table 4.8.4-3. Data Pool Access Configuration Parameters for Rollup Scripts

Parameter Name	Description
DBUSERNAME	The user name for the Database connection (EcDlFtpRollupStats).
DBSERVER	The name of the SQL server for this DB connection. (x4dbl03)
DBNAME	The name of the database you intend to connect to. (ecs)
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.
FTP_LOG_PATH	The path and file name for the default wu-ftp/apache-httpd access log. This parameter is valid for EcDlWuFtpRollup.CFG and EcDlHttpRollup.CFG.
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".

4.8.4.4 Databases

The Data Pool Access Statistics utility uses the Postgresql sedrver.

4.8.4.5 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 4.8.4-4 describes the modules required to run the rollup scripts.

Table 4.8.4-4. Data Pool Access Special Modules

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

4.8.4.6 Outputs

Rollup information is entered in the Data Pool database in the DlGranuleAccess 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).

4.8.4.7 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.

4.8.4.8 Reports

None

4.8.4.9 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 4.8.4.1.3 for details.

4.8.5 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 from the Inventory 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 Inventory database, while the maintenance scripts backup, restore, and delete data in the related Inventory 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.

4.8.5.1 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 Inventory database. Each of these scripts can be run on the command line and connects to the Inventory database to process data contained therein. All of these scripts involve access to the Inventory tables DlGranuleAccess, DlGranuleSubscription, and DlAccessRollup.

Archive Utility – *DlDbArchiveAccessStat.pl*

This script archives data contained in DlGranuleAccess, DlGranuleSubscription, and DlAccessRollup by writing this data to an ASCII file based on an operator-specified time range.

Delete Utility – *DlDbDeleteAccessStat.pl*

This script removes data contained in DlGranuleAccess, DlGranuleSubscription, and DlAccessRollup based on an operator-specified time range.

Restore Utility – *DlDbRestoreAccessStat.pl*

This script restores data archived by the archive utility (contained in the ASCII file) into DlGranuleAccess, DlGranuleSubscription, and DlAccessRollup.

4.8.5.2 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 4.8.5-1 provides a description of these parameters

Table 4.8.5-1. 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 yyyymmdd.
<STOPDATE>	The ending date time range for archiving, deleting, or restoring the data. The format for this parameter is yyyymmdd.
<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>.
<DBUSERNAME>	The database login name.
<DBSERVER>	The host name for the database server
<DBSERVERPORT>	The port number for the database server on the specified host.
<DBNAME>	The name of the RDBMS database (ecs)

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 database login.

4.8.5.3 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.pl <MODE> <STARTDATE> <STOPDATE> <ARCHIVEDIR> <DBUSERNAME>
<DBSERVER> <DBSERVERPORT> <DBNAME>
```

Example:

```
DlDbArchiveAccessStat.pl OPS 20020405 20020505 /home/DBArchive/DataPool/ Labuser01
x4dbl03 5432 ecs
```

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

4.8.5.4 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.pl <MODE> <STARTDATE> <STOPDATE> <DBUSERNAME> <DBSERVER>
<DBSERVERPORT> <DBNAME>
```

Example:

```
DlDbDeleteAccessStat.pl OPS 20020912 20020913 Labuser01 x4dbl03 5432 ecs
```

The above example deletes data in a specified time range.

4.8.5.5 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.pl <MODE> <STARTDATE> <STOPDATE> <ARCHIVEDIR> <DBUSERNAME>
<DBSERVER> <DBSERVERPORT> <DBNAME>
```

Example:

```
DlDbRestoreAccessStat.pl OPS 20020405 20020505 /home/DBArchive/DataPool/ Labuser01
x4dbl03 5432 ecs
```

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

4.8.5.6 Data Access Statistics Main Screen

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

4.8.5.7 Required Operating Environment

The maintenance utilities run on a Linux platform.

4.8.5.7.1 Interfaces to Supporting Products

Table 4.8.6-2 lists the supporting products that these tools depend upon to function properly.

Table 4.8.5-2. Interface Protocols

Product Dependency	Protocols Used	Comments
AIM schema	SQL	Located within the ecs database

4.8.5.8 Databases

The Data Pool Access Statistics utility uses the ecs database AIM schema.

4.8.5.9 Special Constraints

The maintenance utilities run only if the the ECS RDBMS server is running and the “ecs” database is online.

4.8.5.10 Outputs

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

4.8.5.11 Event and Error Messages

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

4.8.5.12 Reports

None

4.8.6 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.

4.8.6.1 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:

> EcDlMostRecentInsert.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 4.8.6-1.

Table 4.8.6-1. 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 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.

4.8.6.2 Most Recent Data Pool Inserts Utility Commands

Examples of how to use this utility is shown below:

1. **EcDlMostRecentInsert.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. **EcDlMostRecentInsert.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.

4.8.6.3 Required Operating Environment

The O/S requirement is Linux 2.x.

4.8.6.4 Interfaces and Data Types

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
Inventory database	SQL	Via SQL server machines

4.8.6.5 Configuration File Format – EcDIMostRecentInsert.CFG

The "config" file contains vital details about how to connect to the database. Without this file, the utility will not run. The config file must be a single-entry plain text ASCII file, with the following format:

```
DBUSERNAME = <string>
DBSERVER= <string>
DBSERVERPORT = <string>
PGM_ID = <string>
DBNAME = <string>
DBSUBSYSTEM = <string>
NUM_RETRIES = <integer>
SLEEP_SEC = <integer>
```

Breakdown of the individual parameters:

Parameter Name	Description
DBUSERNAME	The user name for the database connection.
DBSERVER	The name of the SQL server for this database connection.
DBNAME	The name of the database you intend to connect to
DBSERVERPORT	The port the server is running on
DBSUBSYSTEM	The AIM subsystem
PGM_ID	Program ID used for connecting to the Inventory 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.

4.8.6.6 Special Constraints

EcDlMostRecentInsert utility runs only if the Inventory database server is running and if the database is available. It also assumes the stored procedures are present.

4.8.6.7 Outputs

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

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

Note: The EcDlMostRecentInsert.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.

4.8.6.8 Event and Error Messages

Usage and processing error messages are written to log file.

4.8.6.9 Reports

None

4.8.6.10 Logs

The utility produces a log file called *EcDlMostRecentInsert.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.

4.8.6.11 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.

4.8.6.12 Error Handling

The utility is highly dependent on database server. Connection failure to the database 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 to10 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.

4.8.7 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.

4.8.7.1 Using the EcDlRemap utility

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

>EcDlRemap.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 4.8.7-1 provides a description of these parameters.

Table 4.8.7-1. 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 4.8.7.3 provides some examples along with detailed explanations for executing this utility.

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

The Data Pool Collection-to-Group Remapping utility uses a configuration file, EcDlRemap.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 4.8.7-2 describes the configuration parameters.

Table 4.8.7-2. Configuration Parameters

Parameter Name	Value Description
DBUSERNAME	Database login name for the user of the Inventory database.
DBSERVER	Name of SQL Server hosting Inventory database.
DBSUBSYSTEM	Name of Inventory subsystem.
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.

4.8.7.3 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 Inventory 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
```

4.8.7.4 Required Operating Environment

The Group remapping utility will run on a Linux platform.

4.8.7.5 Interfaces and Data Types

Table 4.8.7-3 lists the supporting products that this tool depends upon in order to function properly.

Table 4.8.7-3. Interface Protocols

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

4.8.7.6 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 Inventory 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 Inventory 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.

4.8.7.7 Outputs

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

4.8.7.8 Event and Error Messages

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

4.8.7.9 Reports

None

4.8.7.10 Logs

The utility produces a log file called EcDlRemap.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.

4.8.7.11 Recovery

The EcDlRemap.pl utility will be able to recover from aborted runs by using the DlRecoveryParameters 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:

```
EcDlRemap.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:

```
EcDlRemap.pl OPS –esdt AST_L1A –version 3 –oldgrp ASTT –newgrp ASTA
```

Then the utility would give a message to the standard output “Recovery detected. EcDlRemap.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.

4.8.7.12 Error Handling

If a database error occurs, you will most likely see the actual 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 that a connection to the Inventory 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.

4.8.8 QA Update Utility (QAUU)

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.

4.8.8.1 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

`-noExitOnError`: 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

4.8.8.2 Required Operating Environment

Linux

4.8.8.3 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
```

4.8.8.3.1 Configuration File Format

Table 4.8.8-1 shows the configuration file parameters.

Table 4.8.8-1. 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 4.8.8-1. 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 4.8.8-1. 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.

4.8.8.3.2 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 incompletion 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 4.8.8.1 Using the QA Update Utility above).

4.8.8.3.3 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"
```

4.8.8.3.4 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.

4.8.8.3.5 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.

4.8.8.3.6 Reports

None

This page intentionally left blank.

4.8.9 Data Pool Move Collection Utility

The Move Collection Utility provides the EED 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 the utility to print extra information to the utility's log file, such as information about when the utility performs specific copy and remove operations. The –debug option causes the utility to print additional troubleshooting information to the log file. The default for these options are non-verbose and non-debug.

The Move Collection Utility also supports fault recovery, 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 AmCollection 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 AmCollection table is updated with the target file system label. Once the AmCollection 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 AmCollection 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 Collection 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 Collection 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 AmCollection 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 REVERB (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 Inventory 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 the AmCollection table may experience an error because Data Pool Ingest cannot locate them. This would result in an operator intervention. **Restart**ing the granule from the operator intervention after the collection move completes successfully will resolve the error.

4.8.9.1 Using the Move Collection Utility

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

> **EcDlMoveCollection.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 4.8.9-1 provides a description of these parameters.

Table 4.8.9-1. Command Line Parameters of the Move Collection 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 4.8.9-1. Command Line Parameters of the Move Collection 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.

4.8.9.2 Move Collection 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.

4.8.9.3 Required Operating Environment

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

4.8.9.4 Interfaces and Data Types

Table 4.8.9-2 lists the supporting products that this tool depends upon in order to function properly.

Table 4.8.9-2. Interface Protocols

Product Dependency	Protocols Used	Comments
aim schema	SQL	Via RDBMS
Perl DBI	DBD::Postgres	Requires proper install of baselined version of Perl.

4.8.9.5 Input File Format

N/A

4.8.9.6 Configuration File Format – EcDlMoveCollection.CFG

The "config" file contains vital details about how to connect to the RDBMS. 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:

```
DBUSERNAME = <string>
DBSERVER = <string>
DBSERVERPORT = <string>
DBNAME = <string>
DBSUBSYSTEM = <string>
PGM_ID = <string>
NUM_DB_RETRIES=<integer>
DB_SLEEP_SEC=<integer>
NUM_DELETE_RETRIES=<integer>
DELETE_SLEEP_SEC=<integer>
```

See Table 4.8.9-3 for a breakdown of individual parameters.

Table 4.8.9-3. Configuration File Parameters

Parameter Name	Description
DBUSERNAME	The user name for the RDBMS connection.
DBSERVER	The host name for the RDBMS server.
DBSERVERPORT	The port for the RDBMS supporting the mode.
DBNAME	The DB name within the RDBMS..
DBSUBSYSTEM	The RDBMS schema/subsystem (aim) hosting this utility.
PGM_ID	Program ID used for connecting to the Inventory 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.

4.8.9.7 Special Constraints

- The Move Collection Utility runs only if the RDBMS is running and if the database is available.

- The Move Collection 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 Collection 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 Collection 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 Collection Utility may not be used to move QA, PH or HDF_MAP collections if they have associated science collections that enable them for ordering. The science collections associated with QA, PH, or HDF_MAP ordering are configured through the Data Pool Maintenance GUI. If QA, PH or a HDF_MAP collection is specified on the command line, the utility will reject the collection move if associated science collections have the collection enabled for ftp or web ordering.

Note: Only one instance of the Move Collection 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**.**

4.8.9.8 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 4.8.9.11.

4.8.9.9 Event and Error Messages

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

4.8.9.10 Reports

None

4.8.9.11 Logs

The utility produces a log file called EcDlMoveCollection<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.

4.8.9.12 Recovery

The Move Collection 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.

4.8.9.13 RDBMS Error Handling

If a RDBMS error occurs, the operator will most likely see the actual RDBMS error string displayed on the screen and in the log. Possible RDBMS 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 RDBMS-sourced error, the utility will immediately stop running.

In the event that a connection to the Inventory 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.

4.8.10 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.

4.8.10.1 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:

EcDlHiddenScramblerDataPool.pl <mode> <command line parameters>

There are four command line parameters that may be used. Table 4.8.10-1 provides a description of those parameters.

Table 4.8.10-1. 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

4.8.10.1.1 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.

```
EcDlHiddenScramblerDataPool.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.

4.8.10.2 Hidden Scrambler Configuration File

The Data Pool Hidden Scrambler utility uses a configuration file, EcDlHiddenScrambler.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 4.8.10-2 describes the configuration parameters.

Table 4.8.10-2. Configuration Parameters

Parameter Name	Value Description
DBUSERNAME	The user name for the RDBMS connection.
DBSERVER	The host name for the RDBMS server.
DBSERVERPORT	The port for the RDBMS supporting the mode.
DBNAME	The DB name within the RDBMS.
DBSUBSYSTEM	The RDBMS schema/subsystem (aim) hosting this utility.
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).

4.8.10.3 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.

4.8.10.4 Required Operating Environment

The Hidden Scrambler Utility will run on a Linux platform.

4.8.10.5 Databases

Table 4.8.10-3 lists the supporting products that this tool depends upon in order to function properly.

Table 4.8.10-3. Product Dependencies

Product Dependency	Protocols Used	Comments
aim schema	SQL	Via RDBMS

4.8.10.6 Special Constraints

The Data Pool Hidden Scrambler utility runs only if the RBDMS is running and if the database is available.

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.

4.8.10.7 Outputs

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

4.8.10.8 Event and Error Messages

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

4.8.10.8 Reports

None

4.8.10.9 Logs

The utility produces a log file called EcDlHiddenScrambler.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.

4.8.10.11 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.

4.8.10.12 RDBMS Error Handling

If a RDBMS error occurs, the actual RDBMS 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 RDBMS-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.

This page intentionally left blank.

4.8.11 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 Inventory database at publication time.

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

- backfill the band information in the Inventory 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 EcDlNewInsertUtilitiyDPAD.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 Inventory 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/EcDlAdHEGStart). 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 Inventory 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 DlInsertActionQueue table in the Inventory 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.

4.8.11.1 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 4.8.11-1 provides a description of these parameters.

Table 4.8.11-1. 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 results in a 'usage' syntax display. The log file for the utility is /usr/ecs/<mode>/CUSTOM/logs/EcDlBandBackfillUtility.log.

4.8.11.2 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.

4.8.11.3 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.

4.8.11.4 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
Inventory database	SQL	Java JDBC invocation of Stored Procedures.
StoreNext client	Proprietary	Exposes the DPL file system on the DPL platform.

4.8.11.5 Input File Format

One granuleId per line.

4.8.11.6 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 EcDlInsertUtility.properties.

4.8.11.7 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.

4.8.11.8 Outputs

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

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 log files in the standard log file location. The log file name is EcDlBandBackfillUtility.log. The verbosity of the log file is controlled by the DEBUG_MESSAGES entry in the EcDlInsertUtility configuration file.

4.8.11.12 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.

4.8.11.13 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.

4.8.12 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 if a manual edit is required in a metadata file. The utility will only replace the granule xml metadata file in the ECS archive. The operator is responsible for making any additionally needed changes to the ECS database, and the granule xml metadata file within the DataPool file systems.

4.8.12.1 Using the XML Replacement Utility

The XRU is started by executing the following:

```
$ EcDsAmXruStart <MODE> [ -xmlfile <filename> | -xmldir <directory> ]
```

The mode parameter is required and either –xmlfile or –xmldir 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.

`-xmldir <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 stornext smallfile/XML archive) directory. The directory must be visible on the host where the utility is run.

4.8.12.2 Required Operating Environment

Linux

4.8.12.3 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 granuleId in the AmGranule table in the ECS 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 before 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.

4.8.12.4 Configuration File Format

Table 4.8.12-1 shows the configuration file parameters.

Table 4.8.12-1. Configuration File Parameters

Parameter Name	Description
PROGRAM_ID	The ECS Program ID for the XRU user (SYB_USER).
DBSERVER	The host name for the ECS database.
DBSERVERPORT	The port number used to connect to the ECS database.
DBSUBSYSTEM	The subsystem for this utility (always aim). This is used to manage the RDBMS schema path. The utility will append a mode suffix to the schema names are needed.
DBNAME	The name of the ECS database (always ecs).
DBUSERNAME	The username used to connect to and perform queries for the ECS 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.

4.8.12.5 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:

```
f4dpl01{cmshared}114: EcDsAmXruStart DEV04 –xmldir /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.

4.8.13 DPL Cleanup Orphan/Phantom Validation (EcDlCleanupFilesOnDisk.pl)

The Data Pool (DPL) Cleanup Orphan/Phantom Validation (EcDlCleanupFilesOnDisk.pl) utility 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 ECS Database (orphans). In addition it will create output files specifying any granules in the ECS Database whose files are missing from the Data Pool disk (phantoms).

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

a. A file in the public or hidden Data Pool directories that does not have a matching entry in the inventory database;
b. A symbolic QA/PH/MP/browse link that does not match a QA/PH/MP/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 AmBrowseOnlineFile 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 AIM schema AmDataFile or AmMetadataFile table where the corresponding file does not exist on the Data Pool disks.
b. A Browse cross reference for a public granule in AmBrowseGranuleXref 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 AmQaGranuleXref 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 AmPhGranuleXref is not represented by a link in the public science directory to the associated PH file.
e. A MP cross reference for a public granule in AmHdfMapGranuleXref is not represented by a link in the public science directory to the associated HdfMap file.

f. 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 file is recorded in the AmBrowseOnlineFile table but the Browse File is missing from the Online Archive disk.

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

a. A public QA/PH/MP file is recorded in the AmDataFile tablebut the file is missing from the Online Archive disk.

b. The public QA/PH/MP 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 Online Archive disks 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_ORDER_AGE) 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.

4.8.13.1 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 4.8.13-1 describes the Configuration Parameters.

Table 4.8.13-1. Configuration Parameters (1 of 2)

Parameter Name	Value Description
DBUSERNAME	The Postgres login name.
DBSERVER	The name of the host running the PostgreSQL Server .
DBSERVERPORT	The port for the RDBMS supporting the mode.
DBNAME	The name of the ECS database (always "ecs").
DBSUBSYSTEM	The name of the subsystem that contains this utility. This controls the schema path used in the RDBMS. The value "aim" should always be used. The mode will be added to any schema names by the utility.
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.

Table 4.8.13-1. Configuration Parameters (2 of 2)

Parameter Name	Value Description
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 AIM database by design. The suggested value for this field is "OUTPUTS BRWS". The MAX_ORDER_AGE parameter applies only to collections in this list.
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.

4.8.13.2 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 Online Archive for the mode. This may take a long time to run, so the DAAC operator could validate a set 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 that 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 4.8.13-2 describes the Command Line Parameters.

Table 4.8.13-2. Command Line Parameters

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 AIM 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.
-fix	No	Allows the user to delete files on disk that are not in the ECS 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.

4.8.13.3 Cleanup Files On Disk command line examples

EcDlCleanupFilesOnDisk.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.

EcDlCleanupFilesOnDisk.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.

EcDlCleanupFilesOnDisk.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.

EcDlCleanupFilesOnDisk.pl TS1 -outputDir mytest

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

EcDlCleanupFilesOnDisk.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 ECS database for the files in a given collection and the associated find command that is run on the Online Archive 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 ECS database. Any Browse no longer referenced by a science granule will be removed from the ECS database. The Browse files in AmBrowseOnlineFile 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 AmBrowseOnlineFile 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.

4.8.13.4 Required Operating Environment

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

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
ECS Database	SQL	The PostgreSQL instance for the mode.

4.8.13.6 Special Constraints

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

4.8.13.7 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>.<yyyymmddhhmmss>

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

For example:

/datapool/DEV04/user/FS1/.orderdata/AMSALlMeVZZf/AE_Land.086PjFMBJRv/2006.07.14/tiny_2sci_2browse.1187370525.76499.RGEN.hdf

/datapool/DEV04/user/FS1/.orderdata/AMSALlMeVZZf/AE_Land.086PjFMBJRv/2006.07.14/tiny_2sci_2browse.1190311192.11894.RGEN.hdf.xml

c.BrowseMissingFiles_dplids_RepairByRestoreOLAFromTape.<processid>.<yyyymmddhhmmss>

-- 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>.<yyyymmddhhmmss>

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

303671
303677
303678

b. **FilesNotInDatabase_files.<processid>.<yyyymmddhhmmss>**

--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.yyyymmddhhmmss: 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/P2061984AAAAAAAAAAAAAA06273091132100_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.h00v09.004.2005312161035.hdf.0220.1193162576.93626.RGEN.hdf

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

/datapool/DEV04/user/FS1/.orderdata/LSR7VEkFZrNF/MODPTQKM.086YkZDjSTj/2005.11.06/MODPTQKM.A2005310.h00v09.004.2005312161035.hdf.0220.1193266902.75079.RGEN.hdf

4.8.13.8 Event and Error Messages

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

4.8.13.9 Logs

A single log named EcDlCleanupFilesOnDisk.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.

4.8.13.10 Recovery

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

4.8.13.11 RDBMS Error Handling

All Data Pool Cleanup components will attempt to deal with RDBMS 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 RDBMS error occurs, the actual RDBMS error string will display on the screen and in the log.

4.8.14 Data Pool Cleanup Granules Utility

The Data Pool Cleanup Granules Utility (EcDlCleanupGranules.pl) was modified for Release 8.1 to provide two basic functionalities for ECS Operations Staff: a mechanism to remove non-ECS granules from the Data Pool, and a way to return ECS granules that failed publication to a known state. Processing both ECS granule IDs and non-ECS granule IDs in a single run of the utility is not supported.

When processing non-ECS granules, the utility will remove the files and database entries associated with the specified granules from the Online Archive and Inventory DB. The relevant command line options for processing non-ECS granule are: (-file, -grans, –expired –limit, –offset, and -theme).

The ability to process ECS granules is provided as a "recovery" mechanism for publication failures that create inconsistent granule states. When processing ECS granules, the utility will set the granule status within the Inventory DB to indicate that the granule is stored the "hidden" Online Archive and also remove the files associated with the granule completely from the Online Archive. After running the EcDlCleanupGranules.pl utility, the ECS granules that were processed can be restored to a "usable" state by next running the EcDlRestoreOlaFromTapeStart utility to copy the files back to the Online Archive from the "tape" archive. If the granule was previously marked as public then events will be generated to communicate the new granule status to ECHO. The relevant command line options for processing ECS granules are: (-file, -geoidfile, and –grans).

4.8.14.1 Data Pool Cleanup Granules Utility Configuration File

The utility uses the shared configuration file EcDlCleanupData Pool.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 4.8.14-1. Configuration Parameters (1 of 2)

Parameter Name	Value Description
DBUSERNAME	The user name for the RDBMS connection.
DBSERVER	The host name for the RDBMS server.
DBSERVERPORT	The port for the RDBMS supporting the mode
DBNAME	The DB name within the RDBMS
DBSUBSYSTEM	The RDBMS schema/subsystem (aim) hosting this utility
DBUSERNAME	The user name for the RDBMS connection.
PGM_ID	Program identifier used as seed to generate database password.

Table 4.8.14-1. Configuration Parameters (2 of 2)

Parameter Name	Value Description
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 DlGranuleExpirationPriorty 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.

4.8.14.2 Using the Data Pool Cleanup Granules Utility

Invocation of EcDlCleanupGranules.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 …>
- -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.

Non-ECS granules may have expiration dates associated with them. To clean up Non-ECS granules in the Data Pool 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 Data Pool Cleanup Granules Utility is:

```
EcDlCleanupGranules.pl  <mode>
                        [-norecovery]
                        [-file <input1 input2 ... inputn>]
                        [-grans <granId1 granId2 .. granIdn>]
                        [-geoidfile <input1 input2 .. inputn>]
                        [-expired]
                        [-offset <# of hours>]
                        [-limit <priority limit>]
                        [-theme <theme1 theme2 .. themen>]
                        [-themexref <theme1 theme2 .. themen>]
                        [-batchsize <batch size>]
```

Table 4.8.14-2. 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 *noprompt* option. Note that the geoid file can contain science granules as well as “non-science” granules.
-expired	No	Cleanup non-ECS granules by expiration date and retention priority.
-offset	No	Use only with the –expired 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 4.8.14-2. Command Line Parameters (2 of 2)

Parameter Name	Required	Description
-limit	No	Use only with the –expired 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 –expired option. Specifies the name of a theme for which cleanup is to be performed. The Cleanup Utility will clean up expired non-ECS granules only if the granules are associated with that theme, and are not associated with any other theme; otherwise the utility will only remove the cross references with that theme. The theme name must be enclosed in quotes (“).
-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.

4.8.14.3 Examples:

Cleanup using a file containing a list of Data Pool granule non-ECS granule ids

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

Restore two granules to a “known state” using a list of Data Pool granule ids

EcDlCleanupGranules.pl TS1 –grans 30987 90876

Cleanup nonECS granules that expired before 5am of the current day and that are associated with themes “test1” or “test2” by combining several options

EcDlCleanupGranules.pl TS1 –expired –offset 5 -limit 200 –theme “test1” “test2”

Cleanup using a file containing a list of geoids

EcDlCleanupGranules.pl TS1 –geoidfile /home/abc/geoidfile1 /tmp/geoidfile2

4.8.14.4. Outputs

The EcDlCleanupGranules.pl script does not produce output files.

4.8.14.5. 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.

4.8.14.6 RDBMS Error Handling

All EcDlCleanupGranules.pl components will attempt to deal with RDBMS 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 RDBMS error occurs, the actual database error string will display on the screen and in the log.

4.8.14.7 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.

4.8.15 Link Checker Utility (EcDlLinkCheck.ksh)

The EcDlLinkCheck.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/EcDlLinkCheck.log" and output all broken links.

Table 4.8.15-1. 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.
-forcefix	No	Specify instead of –fix to indicate that you do not wish the utility to check to see if the file system that the link points to is available. This is useful for fixing links that point to invalid file systems.

4.8.15.1 Using the EcDlLinkCheck.ksh Utility

EcDlLinkCheck.ksh <DIRECTORY_TO_START_CHECK> [-fix]

Examples:

EcDlLinkCheck.ksh /datapool/TS2/ –fix

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

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

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

EcDlLinkCheck.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.

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

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

4.8.15.2 DataPool Link Check Configuration File

None

4.8.15.3 DataPool Link Check Log Files

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

4.8.15.4 DataPool Link Check Output Files

EcDlLinkCheck.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.

4.8.16 AIM Tape Archive Consistency Checking Utility (EcDsAmArchiveCheckUtility)

The AIM Tape Archive Consistency Checking Utility (**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.

4.8.16.1 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 4.8.16-1 shows the command line parameters for the EcDsAmArchiveCheckUtility.

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

Parameter Name	Mandatory	Description
-lo (OutputFilesPath)	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 DB Server – for the DAACs this should be

< n | e | l >4dbl03_srvr.

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 database 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.

4.8.16.2 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:

Table 4.8.16-2. Output Report Files of the EcDsAmArchiveCheckUtility

Report File Name	Content Description
<MODE>.yyyy.mm.dd_hh:mm:ss_<host>.dupGrans	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. This file does not report DFAed granules which don't have files in the 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 InArch	Files are present in the archive but are associated with DFAed granules in the database. This separate report is generated for two reasons: 1). We want to distinguish the situations when granule is simply missing from Db and when granule is present but DFAed (Maybe due to granule deletion failure). 2). DAACs sometimes use DFA flag to hide huge number of granules. The "false" inconsistency could be overwhelming. By putting it in a separate report, DAACs could be more focused on dealing with different inconsistencies.
<MODE].yyyy.mm.dd_hh:mm:ss_<host>.inDbDFAed NotInArch	Files associated with DFAed granules are not in the archive. It is normal that a DFAed granule doesn't have the archive file. This separate report is generated to give an explanation when the "Archive Total" and "DB Total" counts are not consistent due to the DFAed granules.

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
- the number of files that are present in the archive but are associated with DFAed granules in the database
- the number of files that are associated with DFAed granules in the database and are not in the archive
- 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.

4.8.16.3 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 4.8.16-3.

Table 4.8.16-3. Individual Configuration Parameters

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

This page intentionally left blank.

4.8.17 XML Archive Metadata Check Utility (EcDsCheckXMLArchive.pl)

The XML Archive Metadata Check Utility (EcDsCheckXMLArchive.pl) utility checks the entire contents of the XML Archive against the list of XML metadata files contained in the AmMetadataFile / DsMdXMLPath tables within the Inventory database. Note that XML metadata files are only stored for science granules.

4.8.17.1 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 4.8.17-1 shows the command line parameters for EcDsCheckXMLArchive.

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

Parameter Name	Mandatory	Description
-mode	Yes	The mode to be processed.
-user	Yes	The PostgreSQL login for the utility to use to connect to the database.
-server	Yes	The name of the host running the PostgreSQL server that contains the Inventory database.
-serverPort	Yes	The port number for the Postres instance supporting the mode..
-database	Yes	This controls the schema to use for the utility. The value “aim” should always be provided. The utility will make sure the “mode” suffix is added to the schema when necessary.
-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.

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

Parameter Name	Mandatory	Description
-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.

4.8.17.2 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 AmMetadataFile table that are missing files in the XML archive. The directory where the file should be located is indicated by the ArchivePathId column; this is stored in the AmMetadataFile table and it refers to the ArchivePathId within the DsMdXMLPath table. The body of this report file uses the following format:

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

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: <ArchivePathId>  <the fully qualified path where the file was found>
GranuleId          XML File
<GranuleId>                    <XML File Name from AmMetadataFile>
```

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.

4.8.18 Data Pool Cloud Cover Utility (EcDlCloudCoverUtilityStart)

This utility is obsolete in 8.2.

This page intentionally left blank.

4.8.19 DPL XML Check Utility

The DPL XML Check utility provides a mechanism by which the ECS Operations Staff can periodically check for corruption of the XML files in datapool.

In order to detect corruption, the utility verifies the contents of the files are well formed using xmllint.

4.8.19.1 Using the DPL XML Check Utility

The DPL XML Check utility should be started by the user cmshared (or similar). The XML Check utility is started by entering the following command:

EcDlXcu.pl <mode> <command line parameters>

There are seven command line parameters that may be used. Table 4.8.19-1 provides a description of those parameters.

Table 4.8.19-1. Command Line Parameter

Parameter Name	Required	Description
days	No	Optional parameter to specify days since last checked.
percent	No	Optional parameter to specify percentage of files to check
ESDT	Yes, if granuleid or file parameters are not present	Parameter to specify which ESDTs to check. This is a comma separated list (no spaces). Can also specify “ALL” to include all ESDTs.
startdate	No	Optional parameter used with –ESDT option. Specifies starting insert date to use for ESDTs.
enddate	No	Optional parameter used with –ESDT option. Specifies ending insert date to use for ESDTs.
granuleid	Yes, if ESDT or file parameters are not present	Parameter to specify which granules to check. This is a comma separated list (no spaces).
file	Yes, if ESDT or granuleid parameters are not present	Parameter to specify which granules to check. Granule ids should be listing in a file separated by newlines.
outputDir	No	Parameter to specify directory for error files under /workingdata/emd/<MODE>/DplXcu

4.8.19.1.1 DPL XML Check Utility Command Line Examples

1. For an "ESDT" run:

```
EcDlXcu.pl OPS -ESDT ALL
```

The DPL XML Check Utility will perform checking for all xml files in OPS mode

```
EcDlXcu.pl OPS -ESDT AST_L1A.003,MOD29.005 -startdate Jan 20 2008 -
enddate Dec 1 2008
```

The DPL XML Check Utility will performed checking for all AST_L1A.003 and MOD29.005 xml files whose granules have been inserted between Jan 20 2008 and December 1 2008.

```
EcDlXcu.pl OPS -ESDT AST_L1B.003 -percent 50 -days 10
```

The DPL XML Check Utility will perform checking for 50% of AST_L1B.003 granules which have not been checked in the last 10 days.

2. For a "granuleid" run:

```
EcDlXcu.pl OPS -granuleid 22083,22085,22087
```

The DPL XML Check Utility will perform checking for the xml files related to granules 22083, 22085, and 22087in OPS mode

3. For a "file" run:

```
EcDlXcu.pl OPS -file granuleids.txt
```

The DPL XML Check Utility will perform checking for the xml files related to granules specified in granuleids.txt

4.8.19.2 DPL XML Check Configuration File

The DPL XML Check utility uses a configuration file, EcDlXcu.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 4.8.19-2 describes the configuration parameters.

Table 4.8.19-2. Configuration Parameters (1 of 2)

Parameter Name	Value Description
DBUSERNAME	The Postgres DBMS login name.
DBSERVER	The host name for the RDBMS.
DBNAME	The name of ECS database.
DBSUBSYSTEM	The name of the subsystem for this utility (aim). This is used to manage the schema path within the RDBMS.

Table 4.8.19-2. Configuration Parameters (2 of 2)

Parameter Name	Value Description
PGM_ID	Program identifier used as seed to generate database password.
NUM_RETRIES	Number of times database operation will be attempted.
RETRY_INTERVAL	Number of seconds between retries.
MAX_CONCUR_CHECKS	Number of concurrent calls to xmllint that will be allowed.
VALIDATION_OUTPUT_DIR	The default directory to place error output files. The directory should be readable/writeable by cmshared. The suggested directory is /workingdata/emd/<MODE>/DplXcu

4.8.19.3 DPL XML Check Utility Main Screen

The XML Check Utility does not have a main screen. It has a command line interface only.

4.8.19.4 Required Operating Environment

The DPL XML Check Utility will run on a Linux platform.

4.8.19.5 Databases

Table 4.8.19-3 lists the supporting products that this tool depends upon in order to function properly.

Table 4.8.19-3. Product Dependencies

Product Dependency	Protocols Used	Comments
ecs database	SQL	The script uses the “aim” schema within the database.

4.8.19.6 Special Constraints

The DPL XML Check utility runs only if the ECS database server is running and if the database is available. It also assumes all RDBMS functions are present.

4.8.19.7 Outputs

Output of update events and errors will be always appended to a single log file. The DPL Xcu will also produce a phantom file(DPLPhantomXMLs_ecsids.<pid>.<date>) and a failed file(DPLFailedXMLCheck_ecsids.<pid>.<date>). They will be placed in /workingdata/emd/<MODE>/DplXcu. This directory may be further extended using the –outputDir command line option.

4.8.19.8 Event and Error Messages

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

4.8.19.9 Reports

None

4.8.19.10 Logs

The utility produces a log file called EcDlXcu.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.

4.8.19.11 Recovery

The DPL XML Check Utility can recover from interruptions caused by situations such as the system faults or database errors leaving all or some of the xml files unchecked. To recover, the operator would need to specify the –days parameter and enter the number of days since the last time the utility was run. This will ensure xml files which have already been checked are not rechecked.

4.8.19.12 RDBMS Error Handling

If a RDBMS error occurs, the actual RDBMS 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 function stored in the RDBMS. In the event of a RDBMS-sourced error, the utility will immediately stop running.

In the event that a connection to the ECS 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 RETRY_INTERVAL 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.

4.8.20 AIM Map Generation Server (MGS)

The AIM Map generation ServerServer (MGS) provides a mechanism by which the ECS Operations Staff can perform map generation for files in the AIM. MGS retrieves a list of granules to checksum from the AIM database AmMgRequest table. These granules are entered via either the EcAmInsertMapGenerationRequest.pl script (See Section 4.8.23 Map Generation Utility (MGU) for details) or via DPL Ingest. The MGS is able to efficiently generate map files for files in the DataPool, triggered via the records in the AmMgRequest table.

- MGS searches for records in AmMgRequest that has a Status of “N”, and for each GranuelId (granuleId equivalent) a map generation is done. Upon successful map generation, MGS will update the time when map generation took place for each file in the AIM database. MGS will also mark the requests in AmMgRequest table as complete by setting the Status field to “C”.
- Upon detection of map generation failure, the utility will log detailed information about the failure which will include granule ID, ESDT, insert time, complete file path and file name, along with the checksum information. The primary use case of MGS is to run as a service in the background, performing map generations as it finds new map generation requests in the AmMgRequest table, which are currently entered via a user running the EcAmInsertMapGenerationRequest.pl script from the command line.
- Following the standard of most EMD servers, the MGS will not allow multiple instances of itself to run in a particular mode.

4.8.20.1 Using the AIM Map generation Server

The DPL Ingest Processing Service software will call a stored procedure to put a map granule creation request into AmMgRequest table (which acts as MGS’s action queue) in AIM Database for the incoming science granule whose data type is configured to have map files generated. This stored procedure call takes place after the incoming granule is successfully ingested (which means after successful inventory insert). This stored procedure call will only take place if the ingested granule is a science granule. The stored procedure checks the map granule creation flag of the data type of the science granule. If the value of the flag is "N", the procedure does nothing; If the value of the flag is "Y", the procedure inserts a map granule creation request in AmMgRequest table.

Once the above steps have been completed, the user may Run the AIM Map generation Server. The AIM Map generation Server should be started by the user cmshared (or similar). The AIM Map generation Server is started by entering the following command:

EcAmMGSStart <mode>

There is one command line parameter that may be used. Table 4.8.20-1 provides a description of the parameter.

Table 4.8.20-1. Command Line Parameter

Parameter Name	Required	Description
Mode	yes	Mode in which the Map generation Server should be ran.

When running EcDlMGSStart without any parameter on the command line, it will print out its usage, as follows:

Usage: EcAmMGSStart MODE [param1 value1] [param2 value2]...

(NOTE: any command line parameters following the mode are ignored by the server)

4.8.20.1.1 AIM Map generation Server Command Line Examples

To generate map files for all science files associated with the granuleId's found in AmMgRequest table with Status='N' run:

```
EcAmMGSStart OPS
```

The AIM Map generation Server will perform map generation for all granule files associated with the granuleId's found in the AmMgRequest table with Status='N'.

4.8.20.2 AIM Map generation Server Configuration File

The AIM Map generation Server uses a configuration file: EcDsAmMapGenerationServer.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 and are updated manually using a text editor. Table 4.8.20-2 describes the configuration parameters.

NOTE: Since MGS inherits most of its basic functionality from the Checksum Verification Server, some of the configuration parameters will keep the original server's name.

Table 4.8.20-2. Configuration Parameters

Parameter Name	Description
AppLogSize	Maximum size that the application log can be before starting a new application log
AppLogLevel	Level of detail to be logged in the application log
DebugLevel	Level of detail to be logged in the debug log
DBSERVER	Postgres server name for the AIM database connection
DBNAME	AIM database name (ex: AIM_TS1...)
DBUSER	The user name for the AIM database Postgres connection.
DPL_PASSWDSEED	Seed used to dynamically generate a DB login password for the server
DPL_MinPoolSize	The minimal size of the DPL database connection pool.
DPL_MaxPoolSize	The maximum size of the DPL database connection pool.
DPLMapGenerationServiceHostPort	Host Port that the MGS reads from
DPLMapGenerationServiceHosts	Hosts that the server may be ran on
DlCMapGenerationRequestRetryLimit	How many times to retry on a "retryable" error (such as DB connection failure etc…) – default is 5
DlCMapGenerationRequestRetryInterval	How long to wait before retrying a failed operation

4.8.20.3 AIM Map generation Server Main Screen

The AIM Map generation Server does not have a main screen. It has a command line interface only.

4.8.20.4 Required Operating Environment

The AIM Map generation Server will run on a Linux platform.

4.8.20.5 Databases

Table 4.8.20-3 lists the supporting products that this tool depends upon in order to function properly.

Table 4.8.20-3. Product Dependencies

Product Dependency	Protocols Used	Comments
AIM Database	SQL	Via SQL server machines
Inventory Database	SQL	Via SQL server machines

4.8.20.6 Special Constraints

The AIM Map generation Server runs only if the AIM database and Inventory database server is running and if the databases are available. It also assumes the stored procedures are present, EcDsMapGenerater is the user it logs into the DB as and is a registered user in both databases and have proper permission to execute the stored procedures.

4.8.20.7 Outputs

MGS generates both an Application and Debug log file (See Section 4.8.20.10 for details). The main "output" of this server is the changes that it makes to the AmDataFile table with regard to the granules' files for which maps are being generated.

4.8.20.8 Event and Error Messages

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

4.8.20.9 Reports

See outputs above.

4.8.20.10 Logs

MGS produces a log file called EcAmMapGenerationDebug.log and a EcAmMapGeneration.ALOG. The log file will reside in the /usr/ecs/<mode>/CUSTOM/logs directory. In addition to logging the completion of a map generation request, when the MGS has finished processing a request, the corresponding entry in the AmMgRequest table will be marked completed by setting the Status field to 'C'.

4.8.20.11 Recovery

If the AIM Map generation Server goes down during processing, when the operator brings the server backup it will read the AmMgRequest table and see that there is a request who's status is not complete 'C' and continue to process that request.

4.8.20.12 Postgres Error Handling

If a Postgres error occurs, the actual Postgres error string will most likely be logged 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 Postgres-sourced error, the utility will not be able to process any granules.

In the event that a connection to the AIM 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, AmMapGenerationRequestRetryLimit was set to 3 and AmMapGenerationRequestRetryInterval 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.

4.8.21 AIM Map Generation Utility (MGU)

The AIM Map generation ServerServer (MGS) provides a mechanism by which the ECS Operations Staff can perform map generation for hdf4 files in the AIM. MGS retrieves a list of granules to generate maps for from the AIM database AmMgRequest table, which acts as MGS' queue. One of the ways in which the request queue is populated is via the EcAmInsertMapGenerationRequest.pl script. The MGU is able to efficiently generate map files for files in the DataPool, triggered via the records in the AmMgRequest table. (See Section 4.8.22 AIM Map Generation Server (MGS) for instructions on how to run the server).

4.8.21.1 Using the AIM Map Generation Utility

In order to trigger the map generation process by the MGS, the user must first populate the AmMgRequest table with map generation requests. This is done by running the EcAmInsertMapGenerationRequest.pl script (also referred to as MGU or the Map Generation Utility). To run this script, do the following:

1 Log in at the machine where the AIM Map Generation Service is installed (e.g., x4dpl01).

2 To change to the directory for starting the AIM Map Generation Service, type **cd /usr/ecs/<*MODE*>/CUSTOM/utilities** and then press the **Return/Enter** key.

- The working directory is changed to **/usr/ecs/<*MODE*>/CUSTOM/utilities**.

3 At the prompt, type the command to start the AIM Map Generation Utility, in the form of the following.

EcAmInsertMapGenerationRequest.pl –mode <MODE> [-file|f filename[-demo] [-oputputDir] [-granule|g id] [-replace] [-esdt esdtname] [-profile name] [-insertTime [start date] [end date]] [-filesystem|fs dplfilesystem] [-priority priority] [-batchlabel batch] [-help|h]

Example of AIM Map Generation Service Client:

- **EcAmInsertMapGenerationRequest.pl -mode OPS -g 12345**
Generate Map file for AIM granule 12345 in OPS mode

- **EcAmInsertMapGenerationRequest.pl -mode OPS -esdt AST_L1A.004**
Generate Map file for all granules from collection AST_L1A.004 in OPS mode

- **EcAmInsertMapGenerationRequest.pl -mode OPS –file /home/cmshared/test.dat**
Generate Map file for all granules specified in file /home/cmshared/test.dat

Once the above steps have been completed, the user may run the AIM Map Generation Server. The AIM Map Generation Server should be started by the user cmshared (or similar). The AIM Map Generation Server is started by entering the following command:

EcAmMGSStart <mode>

There is one command line parameter that may be used. Table 4.8.21-1 provides a description of the parameter.

Table 4.8.21-1. Command Line Parameter

Parameter Name	Required	Description
Mode	yes	Mode in which the AIM Map Generation Server should be ran.
file|f [filename]	no	Name of a file containing a list of granule IDs to generate maps for
granule|g [GranuleId]	no	A specific granule ID to generate a map for
replace	no	Override any existing maps for a granule IDs who's maps are being regenerated
esdt [ESDTName]	no	Generate maps for all granules belonging to a specific ESDT
insertTime start date end date	no	Generate maps for all granules having science files inserted during the pass in time interval
Demo	No	Demo mode - results are not committed to the AIM database, but are written to a log file, intended for testing and Stand-alone
outputDir	No	Where you want your map files to be placed when in demo mode
help|h	no	Get help on usage/command line options

When running EcDlMGSStart without any parameter on the command line, it will print out its usage, as follows:

> Usage: **EcAmInsertMapGenerationRequest.pl –mode <MODE> [-file|f filename] [-granule|g id] [-replace] [-esdt esdtname] [-profile name] [-insertTime [start date] [end date]] [-filesystem|fs dplfilesystem] [-priority priority] [-batchlabel batch] [-help|h] [-demo] [-outputDir]**

4.8.21.1.1 AIM Map Generation Server Command Line Examples

Example of AIM Map Generation Service Client:

- **EcAmInsertMapGenerationRequest.pl -mode OPS -g 12345**
  Generate Map file for AIM granule 12345 in OPS mode

- **EcAmInsertMapGenerationRequest.pl -mode OPS -esdt AST_L1A.004**
  Generate Map file for all granules from collection AST_L1A.004 in OPS mode

- **EcAmInsertMapGenerationRequest.pl -mode OPS –file /home/cmshared/test.dat**
  Generate Map file for all granules specified in file /home/cmshared/test.dat

4.8.21.2 AIM Map Generation Utility Configuration File

The AIM Map Generation Utility uses MGS' configuration file: EcDsAmMapGenerationServer.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 4.8.21-2 describes the configuration parameters.

Table 4.8.21-2. Configuration Parameters

Parameter Name	Description
AppLogSize	Maximum size that the application log can be before starting a new application log
AppLogLevel	Level of detail to be logged in the application log
DebugLevel	Level of detail to be logged in the debug log
DBSERVER	Postgres server name for the AIM database connection
DBNAME	AIM database name (ex: AIM_TS1..)
DBUSER	The user name for the AIM database Postgres connection.
DPL_PASSWDSEED	Seed used to dynamically generate a DB login password for the server
AIM_MinPoolSize	The minimal size of the AIM database connection pool.
AIM_MaxPoolSize	The maximum size of the AIM database connection pool.
AIMMapGenerationServiceHostPort	Host Port that the MGS reads from
AIMMapGenerationServiceHosts	Hosts that the server may be ran on
DsMapGenerationRequestRetryLimit	How many times to retry on a "retryable" error (such as DB connection failure etc…) – default is 5
DsMapGenerationRequestRetryInterval	How long to wait before retrying a failed operation

4.8.21.3 AIM Map generation Utility Main Screen

The AIM Map Generation Utility does not have a main screen. It has a command line interface only.

4.8.21.4 Required Operating Environment

The AIM Map Generation Utility will run on a Linux platform.

4.8.21.5 Databases

Table 4.8.21-3 lists the supporting products that this tool depends upon in order to function properly.

Table 4.8.21-3. Product Dependencies

Product Dependency	Protocols Used	Comments
AIM Database	SQL	Via SQL server machines
Inventory Database	SQL	Via SQL server machines

4.8.21.6 Special Constraints

The AIM Map Generation Utility runs only if the AIM database and Inventory database server is running and if the databases are available. It also assumes the stored procedures are present, EcDsMapGenerater is the user it logs into the DB as and is a registered user in both databases and have proper permission to execute the stored procedures.

4.8.21.7 Outputs

MGU generates a log file (Logs 4.8.21.10 for details). The main "output" of this utility is the changes that it makes to the AmRequest table. MGU will populate this table with new requests for MGS.

4.8.21.8 Event and Error Messages

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

4.8.21.9 Reports

See section 4.8.2.7 Outputs.

4.8.21.10 Logs

MGS produces a log file called **EcAmInsertMapGenerationRequest**.log. The log file will reside in the /usr/ecs/<mode>/CUSTOM/logs directory. The primary focus of this log is to report the success/failure of the bcp action, which populates AmMgRequest table (MGS's request queue).

4.8.21.11 Recovery

If MGU fails during processing, once restarted it will reprocess all data and regenerate requests for AmMgRequest table, skipping any requests that are already present in the database.

4.8.21.12 Postgres Error Handling

If a Postgres error occurs, the actual Postgres error string will most likely be logged 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 Postgres-sourced error, the utility will not be able to process any granules.

In the event that a connection to the AIM database cannot be established, the utility will print a Postgres connection error, and the user must re-run the utility once the Postgres connection has been restored.

4.8.22 EOSDIS Service Interface (DataAccess)

The EOSDIS Service Interface (ESI) is an interface that permits access to data and services residing within an ECS archive. Recent enhancements to the ECS system have resulted in virtually all archived data being available online for immediate access. ESI takes advantage of this by providing access not only to the data, but also to services on that data.

When initially installed, the ESI system supports two processing services, HEG and GDAL, which are described in greater detail later in this document. These services will be initially enabled for all collections where AmCollection.ConvertEnabledFlag is set to 'Y'. Note that while many collections, covering a wide array of platforms, instruments, processing levels, etc. have been tested by SDPS, it is suggested that the DAACs perform some additional testing on datasets they intend to expose, and use the Data Access GUI, described later in this document, to disable any datasets or options which are found to be problematic. Any such issues should of course also be submitted to SDPS as trouble tickets for repair. It should be noted also that not all of the file formats exposed by the GDAL tool have been tested and the DAAC should take care to test the file formats they would like to expose.

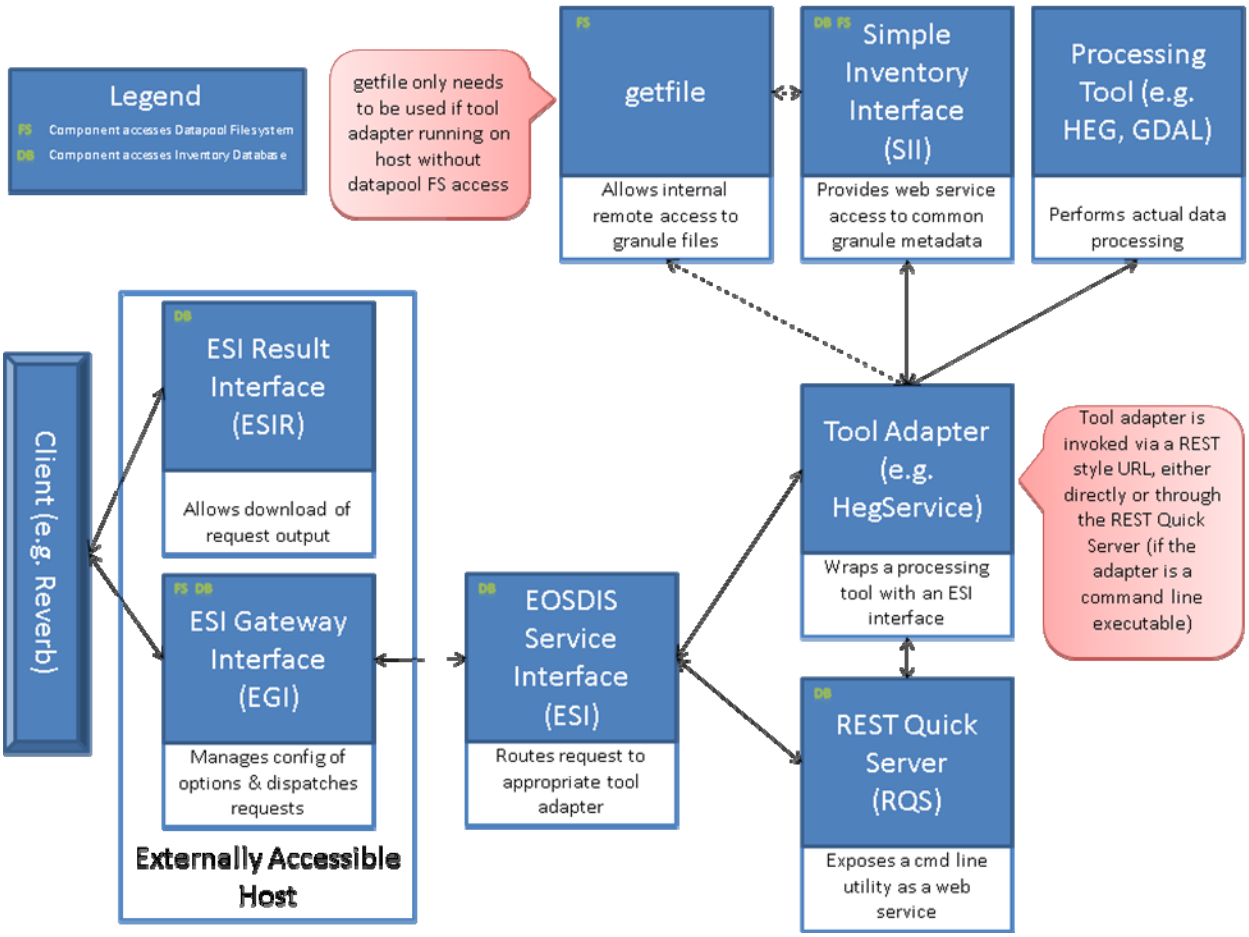


Figure 4.8.22-1. Architectural Overview of the Data Access System

4.8.22.1 Internal Interfaces

The DataAccess system includes a number of components that are for internal use. These components (primarily web applications) run on hosts that are not accessible to the external internet and provide functionality both to support the external facing components as well as to provide functionality to DAAC operators.

4.8.22.1.1 EOSDIS Service Interface (ESI)

The EOSDIS Service Interface (ESI) is a web-service which runs on an internal host (i.e. only accessible on the DAAC internal network). Its primary functionality is to route service requests to subagents, but also includes a test-support inventory access ("drill-down") and service access capability, including the ability to generate echo forms.

4.8.22.1.1.1 Using the EOSDIS Service Interface

The EOSDIS Service Interface (ESI) is a web-service that can be invoked via an HTTP GET or an HTTP POST request. ESI will usually only be called via the ESI Gateway Service (See EGI below). The URL for ESI is as follows:

http://host:port/esi(_MODE)/

To make a capabilities request you can use the following URL:

http://host:port/esi(_MODE)/capabilities/<ShortName.VersionId>

Where <ShortName.VersionId> is the shortname and versionid of the collection you are interested in.

For a processing request you can use the following URL:

http://host:port/esi(_MODE)/processing

The processing parameters are passed in via HTTP GET (ie appending

?param1=value1,param2=value2,... to the URL)

or via HTTP POST. The following is a table of the parameters available:

Table 4.8.22-1. Request Parameters (1 of 2)

Parameter Name	Required	Description
SUBAGENT_ID	Yes	Identification of desired agent to perform internal processing. If this agent is unavaiable or unable to perform the requested processing on the requested file, the request will result in an error. If no value is specified, then ESI will be responsible for choosing a tool to forward the request to.
OUTPUT_DIR	No	Local path of directory where output file should be placed. This parameter is only used if the agent (tool adapter) and the external web API have access to common storage (e.g. a SAN file system).
PROJECTION	No	Output projection for the subsetted file
PROJECTION_PARAMETERS	No	Comma separated name:value pairs of parameters to be used by the output projection. This parameter is only valid if a PROJECTION is also specified.
RESAMPLE	No	The size of pixels in the output file. In the format of [DIMENSION]:[size]
INTERPOLATION	No	The interpolation method to use for rescaling and/or reprojecting of the output data
CLIENT	No	The API used for this request. Valid value are "SSW" and "ESI". This will determine if the server should use SSW or ESI syntax for validating the request. The default value will depend on the particular server, but will need to be "SSW" for any server which intends to be backwards compatible with SSW (since and SSW client will not provide this parameter).
DATASET_ID	No	ECHO DataSetID value (collection title)
START	No	Subset start date/time in RFC-3339 format
END	No	Subset end date/time in RFC-3339 format
BBOX	No	Subset bounding box: sequence of values in decimal degrees, separated by commas, in the order west,south,east,north
SUBSET_DATA_LAYERS	No	Comma-separated variable names
FORMAT	No	Format of subsetted file
NATIVE_PROJECTION	No	Native projection of the subsetted file
OUTPUT_GRID	No	Output grid of the subsetted file
EMAIL	No	E-mail address for asynchronous subsetting response
FILE_IDS	Yes	The granule identifier(s) obtained from the atom:id tags of the granule OpenSearch response. A comma-separated list if there are multiple granules and the granule_multiplicity for the agent is "batch" or "both".

Table 4.8.22-2. Request Parameters (2 of 2)

Parameter Name	Required	Description
FILE_URLS	No	The data URL identifier(s) obtained form the link tags with a rel attribute equal tohttp://esipfed.org/ns/fedsearch/1.0/data#in the granule OpenSearch response. A comma-separated list if there are multiple granules and the granule_multiplicity for the agent is "batch" or "both".

4.8.22.1.1.2 EOSDIS Service Interface Examples

Submit a request for HEG processing of a granule as an HTTP GET request:

http://f4hel01:22500/esi_DEV01/processing?FILE_IDS=249600&CLIENT=ESI&DATASET_ID=MODIS/FTerra Snow Cover Monthly L3 Global 0.05Deg CMG V005&INTERPOLATION=NN&FORMAT=HDF-EOS&SUBAGENT_ID=HEG

Submit the same request for HEG processing of a granule as in the above URL, but as an HTTP POST request:

wget -O - --server-response --post-data 'FILE_IDS=249600&CLIENT=ESI&DATASET_ID=MODIS/Terra Snow Cover Monthly L3 Global 0.05Deg CMG V005&INTERPOLATION=NN&FORMAT=HDF-EOS&SUBAGENT_ID=HEG' http://f4hel01:22500/esi_DEV01/processing'

Notice that in both of the preceding examples there are special characters (e.g. ’ ‘) are should be escaped using HTTP URL escaping rules. In many cases, this escaping is not strictly required, depending on how your HTTP client handles special characters, but it is recommended to escape them to avoid confusion or errors. http://www.w3schools.com/tags/ref_urlencode.asp contains a list of rules for escaping special characters.

Here is the preceding get request properly escaped:

http://f4hel01:22500/esi_DEV01/processing?FILE_IDS=249600&CLIENT=ESI&DATASET_ID=MODIS%2FTerra%20Snow%20Cover%20Monthly%20L3%20Global%200.05Deg%20CMG%20V005&INTERPOLATION=NN&FORMAT=HDF-EOS&SUBAGENT_ID=HEG

Get the ESI DataSet XML that specifies what processing is available for collection AE_SI25.002:

http://f4hel01:22500/esi_DEV01/capabilities/AE_SI25.002.xml

4.8.22.1.1.3 EOSDIS Service Interface Configuration File

The EOSDIS Service Interface uses a properties file, EcDlDaEsi.properties, 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 4.8.22-2 describes the configuration parameters.

Table 4.8.22-2. Configuration Parameters

Parameter Name	Value Description
application.name	Name of this application.
program.id	ID 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.
jdbc.driver.name	The JDBC driver being used.
jdbc.url	The JDBC URL being used.
dpl.database.name	Name of the DataPool Database.
aim.database.name	Name of the AIM Database.
database.user	The username for the database login.
db.crypto.name	Program used for decryption.
sii.url	The URL for the Simple Inventory Interface.
dataaccess.dataset.datacenter	The name of the Data Center which this program is installed.
rqs.path	The path name for the REST Quick Server.

4.8.22.1.1.4 EOSDIS Service Interface Main Screen

The EOSDIS Service Interface contains a built in interface for viewing currently configured service options, building ESI requests, and generating ECHO forms.

4.8.22.1.1.4.1 Inventory Drilldown

The ESI web service provides an internally accessible drilldown functionality that allows the operator to view current archive holdings, obtain automatically generated capabilities description XML and ECHO forms, and submit test requests to ESI. Note that since this interface is meant for testing purposes only, and not for access to external end users, it is not fully featured and not as user friendly as, for instance, the ECHO Reverb interface is.

The front page of the ESI Drilldown can be accessed at

http://x4hel01:22500/esi_<MODE>.

This page contains a number of options to get more information about the ESI system, including API documentation, and example request and capabilities URLs. The most useful feature however, is the link under "Inventory Drilldown", which will take the operator to a list of collections that are enabled for ESI processing.

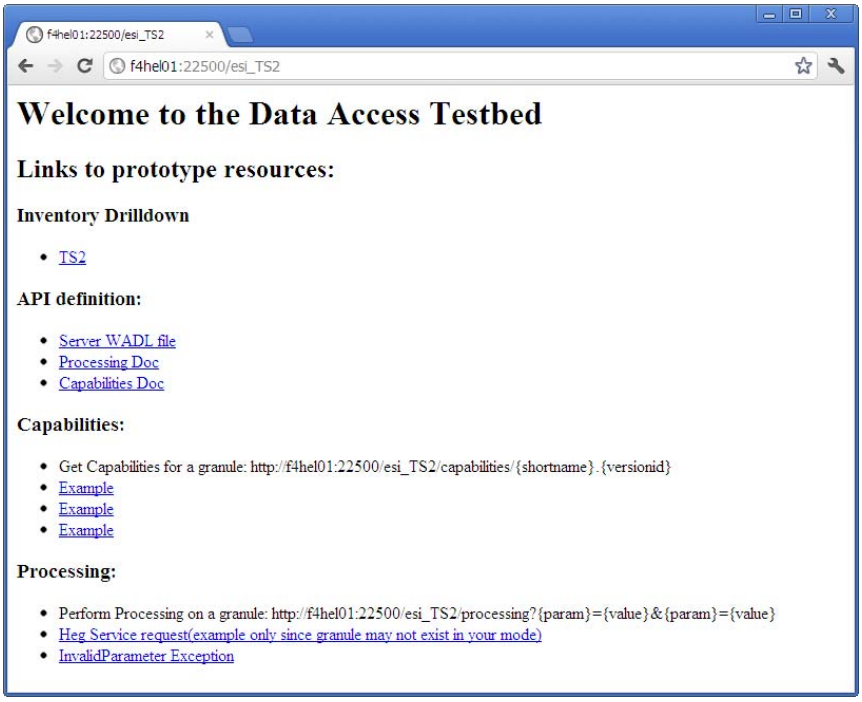


Figure 4.8.22-2. Front Page of ESI Inventory Drilldown

The "HEGable Collections" page list all the collections that have the ConvertEnabledFlag set to 'Y', along with a granule count for each collection obtained from the DlStatESDT table. This count may not always be 100% accurate (it depends on the Datapool Populate Stats script being run regularly) but is much faster than getting an exact count for large collections. Clicking on any collection will bring up further information about that collection.

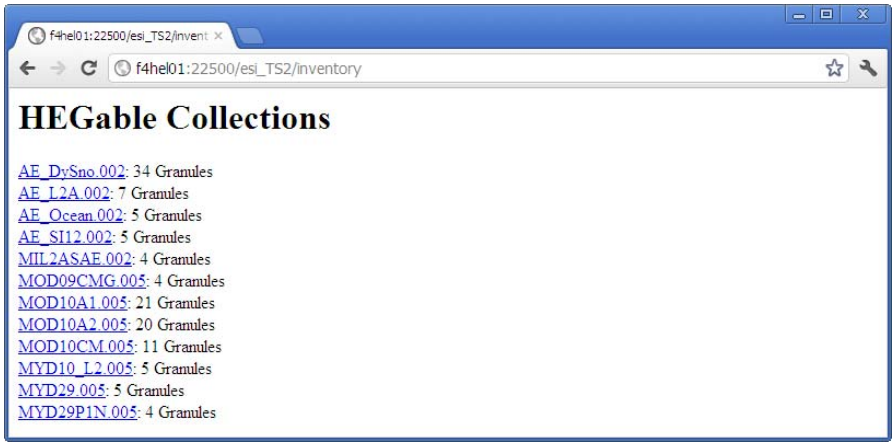


Figure 4.8.22-3. List of Configured ESI Collections and their Granule Counts

The Collection drill-down page provides a list of granules as well as additional data about the collection. In order to prevent the page from being unusable for very large collections, only a maximum of 500 public granules in a collection will be displayed in this page. At the top of the Collection page is a set of links:

- Capabilities: Displays the capabilities XML document, which lists what services and options are available for the collection. This file conforms to http://newsroom.gsfc.nasa.gov/esi/8.1/schemas/ESIDataSets.xsd.
- ECHO Service Form: The ECHO form XML which shall be entered into the ECHO system in order to offer ESI services on a collection via ECHO Reverb. This, and the other ECHO form documents linked to here conform to http://api.echo.nasa.gov/echo/wsdl/EchoForms.xsd.
- ECHO Order Form: Like the above form, however, this form is to be registered in ECHO as an order option form, for use when submitting orders to OMS via EWOC which contain ESI processing. This is not the preferred method of submitting requests (using the ECHO service framework is recommended), but is an option nonetheless.
- ECHO Service Form REST Body: XML conformant to the alpha ECHO REST API (https://api.echo.nasa.gov/echo-rest/) which can be used to upload the service form to ECHO. This is an experimental feature to help facilitate the upload of a large number of forms to ECHO without manual effort. In order to use this XML body, first, acquire a valid ECHO login token (see URL listed above), then save the XML body, and use the instructions in the header of the XML to construct a curl command to perform the upload.
- ECHO Order Form REST body: Same as above, but for the Order, as opposed to service form.

A few notes about ECHO forms:

- When copying ECHO forms XML from the browser to a file, ECHO PUMP, or anywhere else, be sure to use the ‘view source’ feature of the browser and copy from there rather

than from the normal display. Browsers tend to hide some parts of the XML, such as namespace when they format XML files.

- Use firefox when copying this XML. We have encountered problems when using Chrome to view and copy the XML. The file looked the same, but failed when uploaded to ECHO. Chrome must add some sort of hidden character or something. Internet Explorer, and other browsers have not been tested for this purpose.
- ECHO forms can be tested using the ECHO test instance at https://testbed.echo.nasa.gov/reverb/echo_forms/new_test_form. Enter the form XML in the large text box, click 'upload', Make your form selections, and click submit. You will see what the form output would be. This output would then be processed by Reverb (in the case of a service request) or EWOC (in the case of an order), to turn it into an ESI compliant URL.

In addition to the Collection links, this page has a number of links next to each listed granule:

- Capabilities: Same as the collection level capabilities.
- Order form: Simple order form to build an ESI request for the granule.
- ECHO Form XML: Same as the collection level ECHO Service Form.
- Granule Info: Link to the Simple Inventory Interface html page for this granule, which lists basic archive metadata.
- HDF Header Info: Link to the geofile xml page for the granule, which lists hdf header and structural information.

Figure 4.8.22-4. List of Granules for the MOD10CM.005 Collection

The test order form page is dynamically generated based on the configuration for the collection in the capabilities XML mentioned above. This information is in turn derived from the database tables, which are configured via the Data Access GUI. Note that the ECHO forms are also derived from the capabilities XML. The capabilities XML is the definitive listing of dataset capabilities - these other formats are generated by XSLT stylesheets applied to the capabilities XML.

The test order form has a section for each processing tool. A link at the top of the form lets you jump to the section for each tool. The form allows the operator to select the following options while building a request (depending on configuration, some options may not be displayed for each collection):

- Request Type: Synchronous (submit and wait for completion) or Asynchronous (submit and get a request ID and URL to check back later on the request).
- Format: Desired output file format.
- Projection (and Parameters): Desired output projection and parameters.
- Bands: Desired band(s) (if not all bands).
- Spatial Subset: Desired spatial bounds, if performing spatial subsetting.
- Pixel Size: Desired pixel size and dimension for resampling. For HEG, this is currently limited to resizing the image by a percentage only.
- Interpolation Method: The method to use for any interpolation performed as part of the processing.
- Email Address: Email address to sent status updates to.
- Additional GranuleIds: If you want to include more granules in the request, besides the initially selected granule, list their IDs here. NOTE that this only works for asynchronous requests.

Once all of the request options have been selected, scroll to the bottom (or top) of the section for the selected service and click the “Submit request to <Service>” button. This will take you to the submittal page.

The ESI submittal page provides a number of options of how to submit the request (in order to exercise different parts of the system.

- External EGI Request URL (POST): This option most closely matches the normal method or submitting a request from ECHO Reverb. The operator can submit the request either by clicking the submit button or copy/pasting a ‘wget’ command onto the command line. Either way, the request will be sent to the (external facing) EGI service, registered in the AmDaRequest table, and queued (depending on configuration and current load) and processed.
- External EGI Request URL (GET): This option is similar to the above, but submits the request as a GET rather than POST. This option only is available for single granule requests. While this option is not currently exposed via Reverb, it is useful for a more simple, on-demand request since the url can be entered in any browser and, when the

Request Mode is Synchronous, the response to the request will contain a url to download the resulting file(s).

- ESI Request URL (GET): This option will send the request directly to the internal ESI interface. It will not be registered in the database tables, and the resulting files will only be accessible via their native paths, rather than via the ESIR interface.
- EWOC parameter file optionselect line: This option, which is only available for asynchronous requests, gives an example line that could be added to the properties file of the EWOC test driver.

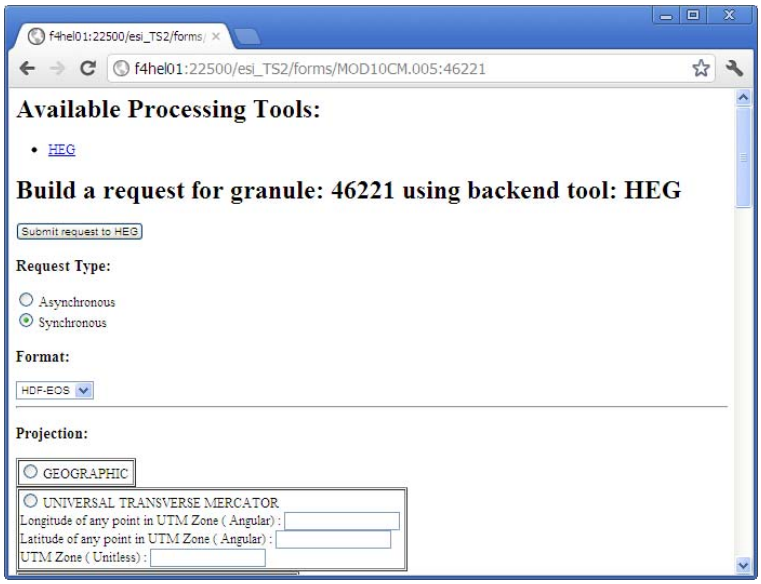


Figure 4.8.22-5. ESI Testing Order Form (Part 1)

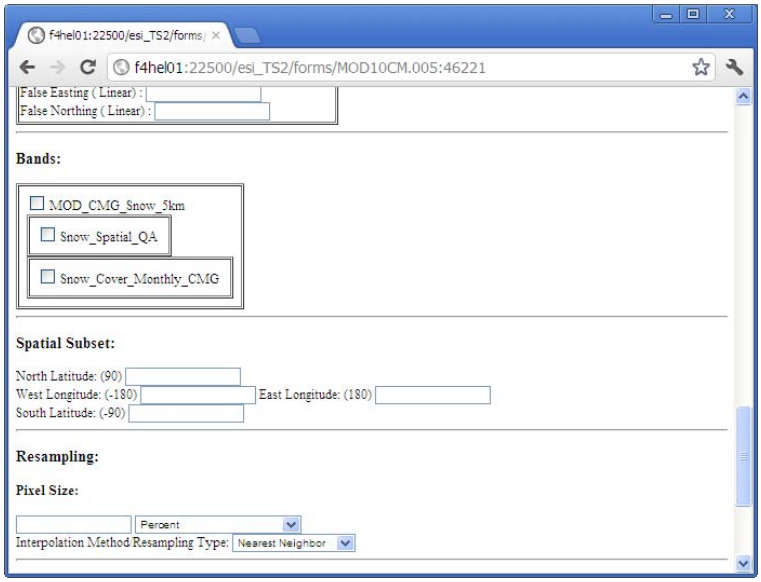


Figure 4.8.22-6. ESI Testing Order Form (Part 2)

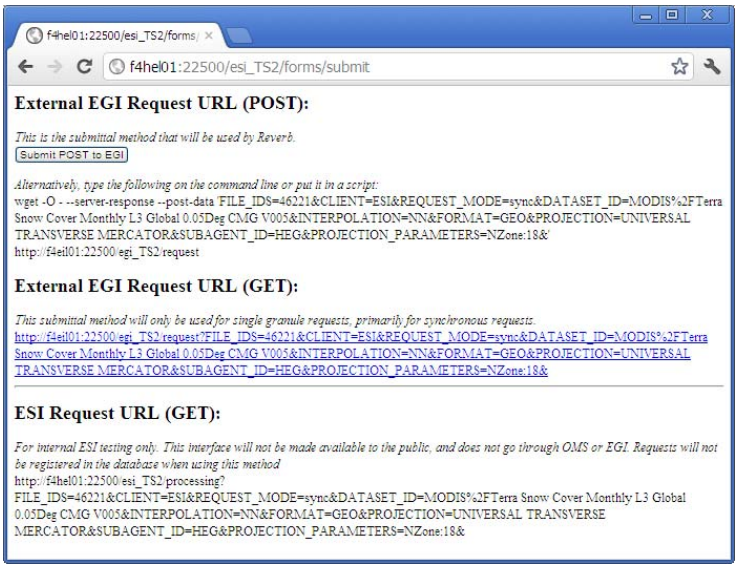
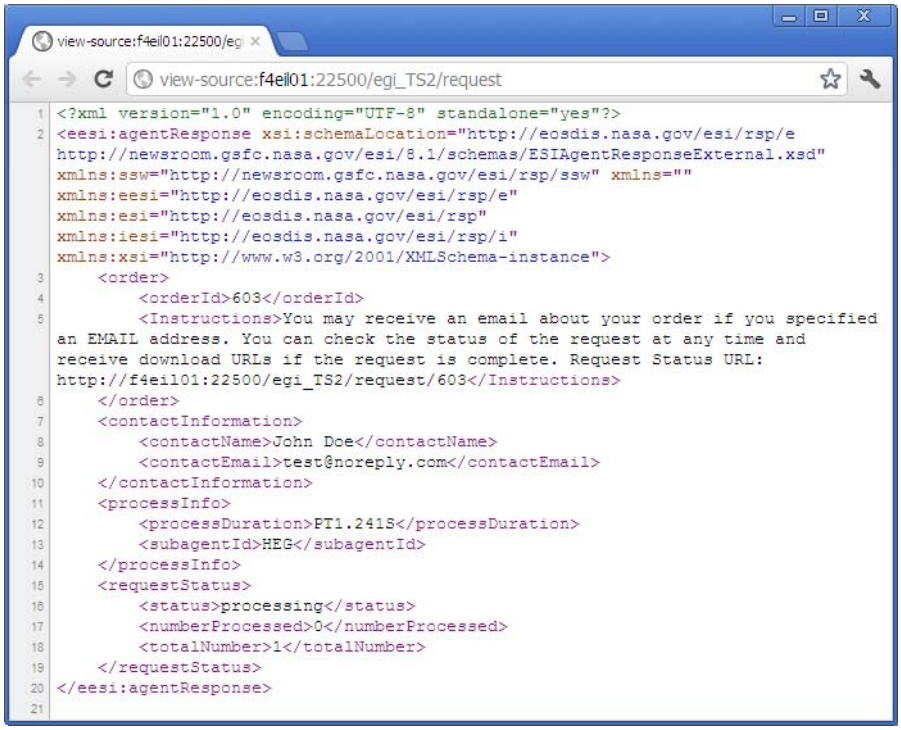


Figure 4.8.22-7. ESI Request Submittal Page (with Multiple Submittal Method Options)

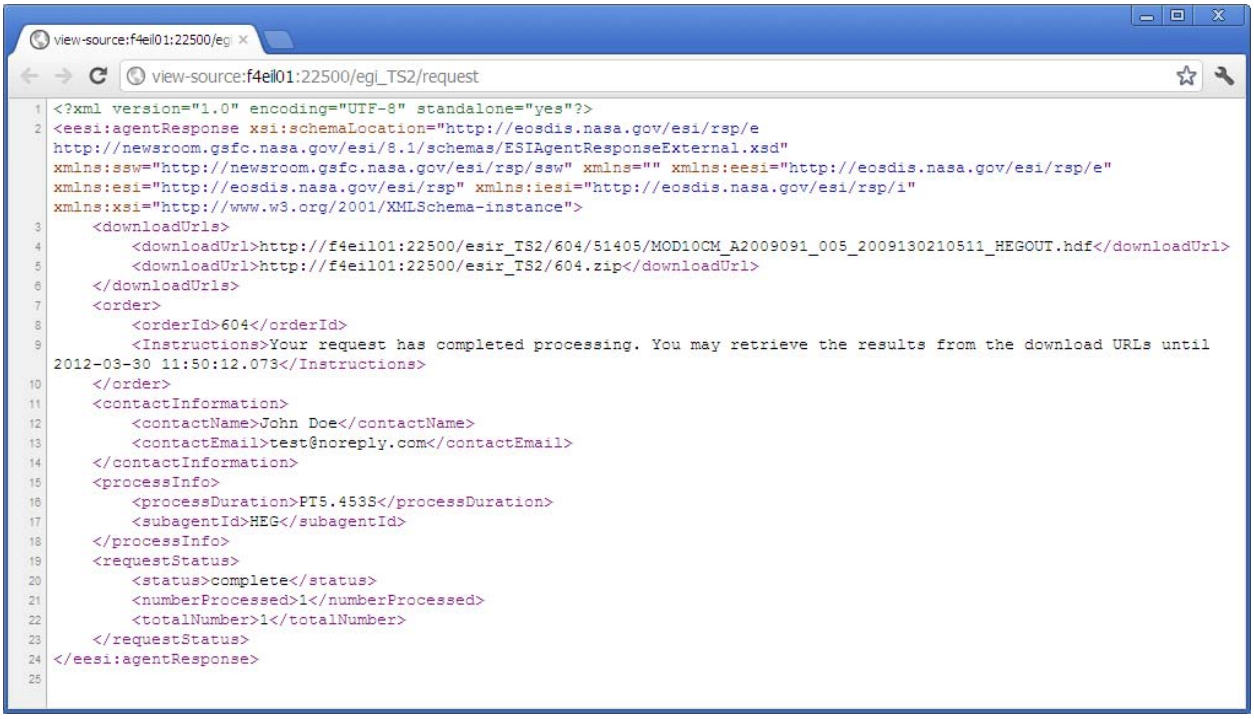
Once you submit a request using one of the methods listed above, you will receive an XML response as shown below. Note that depending on the submittal method and browser being used, the response may appear as unformatted text versus XML. In this case, use the browsers 'view source' feature to view the full XML. Depending on whether the request was synchronous or asynchronous, and how quickly the request was processed, the response will either have an

orderId or a list of download URLs. The XML will also contain a URL to get the current request status. This is especially useful to ping for the progress of an asynchronous request. This feature is used for instance by Reverb in order to provide the user real time status updates in the form of a progress bar. The status response will be in the same format as the initial response to submitting the request.




```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<eesi:agentResponse xsi:schemaLocation="http://eosdis.nasa.gov/esi/rsp/e
http://newsroom.gsfc.nasa.gov/esi/8.1/schemas/ESIAgentResponseExternal.xsd"
xmlns:ssw="http://newsroom.gsfc.nasa.gov/esi/rsp/ssw" xmlns=""
xmlns:eesi="http://eosdis.nasa.gov/esi/rsp/e"
xmlns:esi="http://eosdis.nasa.gov/esi/rsp"
xmlns:iesi="http://eosdis.nasa.gov/esi/rsp/i"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <order>
        <orderId>603</orderId>
        <Instructions>You may receive an email about your order if you specified
an EMAIL address. You can check the status of the request at any time and
receive download URLs if the request is complete. Request Status URL:
http://f4eil01:22500/egi_TS2/request/603</Instructions>
    </order>
    <contactInformation>
        <contactName>John Doe</contactName>
        <contactEmail>test@noreply.com</contactEmail>
    </contactInformation>
    <processInfo>
        <processDuration>PT1.241S</processDuration>
        <subagentId>HEG</subagentId>
    </processInfo>
    <requestStatus>
        <status>processing</status>
        <numberProcessed>0</numberProcessed>
        <totalNumber>1</totalNumber>
    </requestStatus>
</eesi:agentResponse>
```

Figure 4.8.22-8. Example Response from EGI Request Submittal (Request Still in Progress)




```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<eesi:agentResponse xsi:schemaLocation="http://eosdis.nasa.gov/esi/rsp/e
http://newsroom.gsfc.nasa.gov/esi/8.1/schemas/ESIAgentResponseExternal.xsd"
xmlns:ssw="http://newsroom.gsfc.nasa.gov/esi/rsp/ssw" xmlns="" xmlns:eesi="http://eosdis.nasa.gov/esi/rsp/e"
xmlns:esi="http://eosdis.nasa.gov/esi/rsp" xmlns:iesi="http://eosdis.nasa.gov/esi/rsp/i"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <downloadUrls>
        <downloadUrl>http://f4eil01:22500/esir_TS2/604/51405/MOD10CM_A2009091_005_2009130210511_HEGOUT.hdf</downloadUrl>
        <downloadUrl>http://f4eil01:22500/esir_TS2/604.zip</downloadUrl>
    </downloadUrls>
    <order>
        <orderId>604</orderId>
        <Instructions>Your request has completed processing. You may retrieve the results from the download URLs until
2012-03-30 11:50:12.073</Instructions>
    </order>
    <contactInformation>
        <contactName>John Doe</contactName>
        <contactEmail>test@noreply.com</contactEmail>
    </contactInformation>
    <processInfo>
        <processDuration>PT5.453S</processDuration>
        <subagentId>HEG</subagentId>
    </processInfo>
    <requestStatus>
        <status>complete</status>
        <numberProcessed>1</numberProcessed>
        <totalNumber>1</totalNumber>
    </requestStatus>
</eesi:agentResponse>
```


Figure 4.8.22-9. Example Response from EGI Request Submittal (Request Complete)

4.8.22.1.2 HegService

The HDF-EOS to GeoTIFF Converter (HEG) webservice (AKA HegService) is a REST style webservice which wraps the HEG executables and makes them available to the ESI system. The HEG Service is based on the old HEG Server application, but has increased functionality, a new interface, and many bug fixes.

HegService is a Tomcat deployed web application running on the x4hel01 host. This host is only accessible on the DAAC internal network, so any access to it from the outside network (e.g. ECHO) must be mediated by the ESI Gateway Interface (EGI). All interaction with the HEG Service is via HTTP GET requests that contain all request parameters as key/value pairs in the URL query parameters.

HegService accepts requests as URLs according to the ESI Internal API, builds the parameter files that the HEG executables accept and then runs the appropriate executable to fulfill the request. Depending on the nature of the request, the HEG Service may initiate multiple HEG processes in order to satisfy the expected processing. Once processing is completed, HegService returns an XML response conformant to the ESI Internal Response API to the calling entity.

4.8.22.1.2.1 Using the HegService Interface

HegService is a web-service that can be invoked via an HTTP GET request. HegService will usually only be called via the ESI Gateway Service (See EGI below). The URL for HegService is as follows:

http://host:port/HegService(_MODE)

followed by your query string (ie appending ?param1=value1&param2=value2...). The following is a table of the parameters available. Note that either a FILE_URLS or FILE_IDS is required but none of the others.

Table 4.8.22-3. Request Parameters

Parameter Name	Values	Description
FILE_URLS	URL String	The data URL identifier for a granule. Only one URL is allowed for processing.
FILE_IDS	Integer ID (unsigned)	The granule-identifier for a granule, only one allowed. HegService will use this Id, if provided, to attempt to retrieve the file's local access path and the granule's production date for Processing History purposes using SII. If SII is not available, a FILE_URL must be provided. If a FILE_URL and FILE_ID are provided they must correspond to the same granule or an error will be reported and processing stopped.
OUTPUT_DIR	<Path>	Local path of directory where output file should be placed. This parameter is only used if the client machine and the HegService have access to common storage (e.g. a SAN file system).
FORMAT	<Format>	Format of output file – See list of format options
PROJECTION	<Name>	Output projection for the output file. See list of Projections.
PROJECTION_ PARAMETERS	name **:** value **, …**	Parameters to be used by the output projection. This parameter is only valid if a PROJECTION is also specified and can include only the valid parameters for that projection. See list of Projection options.
SUBSET_DATA_ LAYERS	obj **:** field [**:**band[n-m] **: … , …**	See description of Subset_Data_Layer syntax
BBOX	W,S,E,N	Subset area bounding box – a sequence of values, in degrees, ranging +/- 180 east/west longitude, +/- 90 North/South latitude, separated by commas, in the order west,south,east,north
RESAMPLE	<Dimension> **:** <size>	The size of pixels in the output file. If Dimension is PERCENT, the Size is percent of output grid size. Accepted dimensions: Degrees, ArcSec, DMS, Percent. See note about resampling dimensions.
INTERPOLATION	Nearest_Neighbor, Bi-Linear, Cubic_Convolution	The interpolation method to use for rescaling and/or reprojecting of the output data
INCLUDE_META	Yes, No	Directive to include additional meta-information files, including the processing history and the granule meta-information files.

Table 4.8.22-4. Additional Parameters for Test or Debugging Purposes

Parameter Name	Value	Description
TIMEOUT	Integer >= 0 (d,h,m,s)	Allows an override to the default configuration setting. Integer can optionally be followed by d (days), h (hours), m (minutes), s (seconds)
DEBUG	Yes, No	Allows an override to the default configuration setting. When YES, all working files are preserved for debugging purposes; when NO all working files are deleted.
MAX_STITCHED_SOM_BLOCKS	Integer >= 0	Allows an override to the default configuration setting. For MISR granules, HegService will split a job to ensure that no more than this specified number of SOM-Blocks are being stitched together. Zero implies no constraint.

About the use of GetFile or other services in the File_URLs

The FILE_URL parameter is a true URL reference in that a web-resource request will be issued if necessary to retrieve the file to a local directory. A local path is treated as just that (for the machine executing HegService), but a full URL (HTTP:… or FTP:…) is processed by requesting the resource from the network. One example usage is to reference the ESI testbed resource named GetFile running on another machine to retrieve a file from that machine. This would be used for instance if HegService were running on a machine without access to the datapool filesystem.

Note also that HegService has a RemoteGetFileURL configuration parameter that if set, its value is prepended to the given FILE_URL value, thus enabling a GetFile retrieval without having the GetFile service explicitly referenced.

Resampling Dimensions

With this release, HegService introduces a new feature for setting the pixel-size using Percent. HegService will generate a HEG command parameter file that sets the pixel-size as a percent of the input granule's pixel-size. Note, however, that the Percent value given is used to scale the output grid-size, the inverse of the pixel-size. A setting of *resample=percent:80* will result in an output grid with fewer pixels, and for a given area, the pixel-size will be greater.

Other than the use of Percent, however, interpretation and conversion of resampling dimensions, is not performed. This makes the setting of pixel-size somewhat inconvenient since it is necessary to know the expected dimension for the pixel-size, given the input granule and output projection, to set the value correctly. In effect, other than Percent, the dimension specified is ignored and it is up to the user to specify a value that will be interpreted correctly by the HEG conversion utility called.

Note that a dimension specification is required; compliant with the listing above, but other than Percent the dimension designation itself is not considered. The discussion that follows is for users who desire to set the pixel-size explicitly and not using the percent feature.

First, it must be considered if the input granule is a Swath granule. In this case, the pixel-size is always interpreted as decimal degrees, regardless of the output projection specified. Otherwise, for Grid granules (including MISR Grids) if the output projection is specified for a request, then

for Geographic projections the pixel-size is interpreted as decimal degrees, while for all but Geographic output projection, the pixel-size is interpreted as meters.

The output projection in effect is less clear for Grids if the projection is not specified. In this case, the output projection is the same as that of the input granule. (Note that while not particularly relevant to the discussion of pixel-size, since the pixel size for Swath processing is always specified in decimal degrees, Swath processing is also distinct in having a default output projection of Geographic not Swath (the input projection) – there is no Heg capability for generating Swath output files).

If a specific pixel-size setting is required, please note the availability of the SII/GeoFile service, which will provide the native projection and pixel-size of the input granule, thus providing a suitable baseline for specifying a new value.

For expert users familiar with HEG – for grid granules that are processed using HEG's **resample** executable (non-MISR) - Resample requires a pixel size specified in arcsec for Geographic output projection. HegService does perform the conversion from decimal degrees to arcsecs. This is a convenience feature implemented internally by HegService, so that the pixel size may be expressed in the more conventional dimension of decimal degrees.

For this release of EGI (see below) and ESI (see above) the default configuration allows only Percent as a resampling dimension for HegService calls (for collections mapped to a HegService Service configuration). Thus, the Echo/Reverb interface and the ESI Testbed Forms do not offer any other options. If this is of concern there are several points to consider –

- Any collection-service mapping configuration can be modified to accept resampling dimensions that will be passed through to HEG (see DA-Configuration GUI below).
- There is no error handling to ensure a particular dimension is used in correct association with the output projection, but neither will manually inserted alternatives be rejected.
- The ESI Testbed generated calls to HegService may be manually edited to insert alternative resample specifications.
- Direct calls to the HegService may be constructed as well, and the alternative resample specifications inserted
- We anticipate a future release of HegService could address revisions in this area – e.g., to allow "Native" as a dimensions value (for pass-through) or to provide conversion from the specified dimension to the expected dimension as required.

Subset_Data_Layers Syntax

Object **:** Field (**:** Dimension-Name [N_1 (- N_2)]) **:** …

-or- Object **:** Field (**:** N_1 (- N_2)) **:** …

(parenthesis denote optional elements and do not appear in the Subset_Data_Layers, square-brackets, however, are part of the syntax and do appear in Subset_Data_Layers)

The dimension names are optional since, for this release, you must specify the dimensions in the proper order – as they are organized in the data-granules. The band-numbers are the essential items and may be specified as a range (N_1 - N_2), or as a single entry. Object and Field names are both required. Dimensions are optional, but if not specified, all at that level and below are implied.

Of course, the Subset_Data_Layers parameter is optional, with the default being, if not specified, that all objects, fields and bands (dimensions) will be processed.

Valid Values for Formats, Projections and Projection Parameters

Table 4.8.22-4 shows the valid values for Formats. Table 4.8.22-5 shows the valid values for Projections and the associated Projection Parameters.

Table 4.8.22-5. Valid Formats

Code	Description
GEO	Geospatial Tagged Interchange Format
HDF-EOS	Hierarchical Data File - Earth Observational Science

Table 4.8.22-6. Valid Projections and Projection Parameters

Projection	Valid Projection Parameters							
Geographic								
Polar Stereographic	LongPol	TrueScale			SMajor	SMinor	FE	FN
Universal Transverse Mercator	LonZ	LatZ	NZone					
Mercator	CentMer	TrueScale			SMajor	SMinor	FE	FN
Transverse Mercator	CentMer	OriginLat	Factor		SMajor	SMinor	FE	FN
Lambert Azimuthal	CentLon	CenterLat			Sphere		FE	FN
Lambert Conformal Conic	CentMer	OriginLat	STDPR1	STDPR2	SMajor	SMinor	FE	FN
Albers	CentMer	OriginLat	STDPR1	STDPR2	SMajor	SMinor	FE	FN
Sinusoidal	CentMer				Sphere		FE	FN
State Plane			NZone					
Cylindrical Equal Area	CentMer	TrueScale			SMajor	SMinor	FE	FN

Projection Parameters

CENTLON	Center-Longitude
CENTMER	Longitude of Central Meridian
LONGPOL	Longitude descending from Pole along centerline of map
LONZ	Longitude in Zone
CENTERLAT	Center-Latitude
ORIGINLAT	Latitude of Projection Origin
LATZ	Latitude in Zone
TRUESCALE	Latitude, decimal degrees (ddmmss as integer) where scale is true
STDPR1	Std Parallel-1, Latitude (Conic Secant Intersect Latitude 1 of 2)
STDPR2	Std Parallel-2, Latitude (Conic Secant Intersect Latitude 2 of 2)
NZONE	UTM or State-Plane Zone number
FACTOR	Scale-Factor
SMAJOR	Semi-Major-Axis of Ellipsoid model of earth's shape
SMINOR	Semi-Minor-Axis of Ellipsoid, or zero for sphere (default), or Eccentricity-Squared if negative
SPHERE	Radius of Sphere (Earth) or zero => 6,370,997 (default)
FE	False-Easting, +/- Longitude
FN	False-Northing, +/- Latitude

4.8.22.1.2.2 HegService Examples

Submit a request for HEG processing of a granule:

http://f4hel01:22500/HegService_DEV01?FILE_IDS=249600&INTERPOLATION=NN&FORMAT=HDF-EOS

Notice that special characters (e.g. ' ', etc) are typically escaped using HTTP URL escaping rules. In many cases, this escaping is not strictly required, depending on how your HTTP client handles special characters, but it is recommended to always escape them to avoid confusion or errors. http://www.w3schools.com/tags/ref_urlencode.asp contains a list of rules for escaping special characters.

A more involved request for HEG processing of a granule:

http://f4hel01:22500/HegService_DEV01?FILE_IDS=249600&Format=Geo&BBox=-180,0,-50,75&Subset_Data_Layers=MOD_CMG_Snow_5km:Snow_Cover_Monthly_CMG&Projection=Mercator&Projection_Parameters=CentMer:-120&Resample=Percent:75&Include_Meta=yes

4.8.22.1.2.3 HegService Interface Configuration File

HegService uses a properties file, HegServiceConfig.properties, 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 4.8.22-6 describes the configuration parameters. In some cases, when an alternate name is defined, the name that is defined in the ECS-Assist Configuration settings is provided for reference.

Table 4.8.22-7. HegService Configuration Parameters (1 of 2)

Parameter Name	Value Description
application.name	Name of this application (Optional).
log.operations.level	Level of logging desired in operational log: NONE, INFORMATION, VERBOSE or XVERBOSE
log.performance.level	Level of logging desired in performance log: NONE, INFORMATION, VERBOSE or XVERBOSE
log.debug.level	Level of logging desired in debug log: NONE, INFORMATION, VERBOSE or XVERBOSE
log.threshold	Size of log files before new ones are created.
log.overwrite	If true, log file will be overwritten for each run.
log.rotation.number	Number of log files that will be rotated through.
HegService.application. workDirRoot	Required for HegService operations, a directory for working files, sub-directories created within for each request (allows concurrent processing)
HegService.application. tempDirRoot	Optional, a directory for output files being generated. If undefined, a sub-directory to the current working directory will be created.
HegService.application. copyOutput	Optional (default is false) - true / false. If true, and tempDirRoot is set, forces copy of output files after processing (vs. writing to output-dir during processing), if this is better for performance or other reasons. Note that copy-after-processing is always the case if a given request specifies an output directory.
HegService.application. LocalGetFileURL (HegServer_Outfile_GetFile_URL)	Optional – a URL that will be added to the output file specification if users typically don't have access to file-system of the application itself.
HegService.application. RemoteGetFileURL (HegServer_Infile_GetFile_URL)	Optional – a URL that may be used to retrieve the input file if on a different file-system from the application itself, or if copy of the input file to the working directory before processing is preferred.
HegService.application. SIIURL (HegServer_SIIURL)	Required – The URL for the Simple Inventory Interface. Note that the mode for this SII URL should match the mode for the HegService installation.
HegService.application. debugFlag	Required – designates if working files are to be kept for debugging purposes. If false, all working files and directories are deleted.
HegService.application. timeout	Required – HegService will kill HEG processing if its processing time exceeds this value. Specified as a positive integer, optionally followed by a units designator: d – days, h – hours, m – minutes, s – seconds (seconds is assumed if not specified).
HegService.application. maxStitchedSomBlocks	Required – For MISR granules, HegService will split jobs to ensure that no more than this number of SomBlocks will be stitched together. Specified as a positive integer; zero implies no limit and HegService will not split jobs on the basis of SomBlock counts.
REQUEST_CLEANUP _AGE	Required – HegService spawns a separate thread to periodically check for files in workDirRoot and in tempDirRoot that are older than this specified age. These files and directories are then deleted. Specified as a positive integer, optionally followed by a units designator: d – days, h – hours, m – minutes, s – seconds (seconds is assumed if not specified).

Table 4.8.22-7. HegService Configuration Parameters (2 of 2)

Parameter Name	Value Description
REQUEST_CLEANUP _INTERVAL	Required – For the cleanup thread described above, this values determines the time between clean-up processing runs. Specified as a positive integer, optionally followed by a units designator: d – days, h – hours, m – minutes, s – seconds (seconds is assumed if not specified).
CONTACTNAME	Required – For ESI Standard responses, the contact name
CONTACTEMAIL	Required – For ESI Standard responses, the contact e-mail
ESI.DAAC_NAME	Required – For ESI Standard responses, the DAAC site name
ESI.DAAC_URL	Required – For ESI Standard responses, the DAAC site URL

4.8.22.1.2.4 Main Screen

The HegService Interface does not have a main screen. It is a webservice. Please see ESI Usage above.

4.8.22.1.3 GDAL

The Geospatial Data Abstraction Library (GDAL) provides libraries and command line tools that support conversion and processing of a large number of geospatial data formats. More information about GDAL and the data formats and processing it supports can be found at www.gdal.org.

For release 8.1, only a limited subset of GDAL functionality is supported, namely the reformatting capabilities. Additional functionality may be exposed at a later date. You can find the gdal command line tools at '/tools/gdal/bin'. The gdal_translate command is used by the GDAL adapters to perform reformatting. In addition, the gdal_info command is used in various places throughout the Data Access system to obtain header information about granules. More information about these commands and others delivered within the GDAL package, see www.gdal.org.

The build of GDAL that is delivered with 8.1 supports a number of data formats, which are listed in the table below. The GDAL adapter described in this section is initially configured to support reformatting to all of these formats. However, not all formats have been thoroughly tested. Some formats, like GTiff, PNG, BMP, GIF, and JPEG have been shown to work for most datasets. Others are more obscure, and would only be relevant to certain users and certain datasets. DAACs should tailor their configuration to support only those data formats that are relevant to their users and data. In the table below, the 'File Type' column represents the value that the adapter expects in the FORMAT request parameter.

Table 4.8.22-8. Valid Formats (1 of 2)

File Type	Description
VRT	Virtual Raster
GTiff	GeoTIFF
GEO	GeoTIFF
NITF	National Imagery Transmission Format
HFA	Erdas Imagine Images (.img)
ELAS	ELAS
AAIGrid	Arc/Info ASCII Grid
DTED	DTED Elevation Raster
PNG	Portable Network Graphics
JPEG	JPEG JFIF
MEM	In Memory Raster
GIF	Graphics Interchange Format (.gif)
XPM	X11 PixMap Format
BMP	MS Windows Device Independent Bitmap
PCIDSK	PCIDSK Database File
PCRaster	PCRaster Raster File
ILWIS	ILWIS Raster Map
SGI	SGI Image File Format 1.0
SRTMHGT	SRTMHGT File Format
Leveller	Leveller heightfield
Terragen	Terragen heightfield
HDF4Image	HDF4 Dataset
HDF-EOS	HDF4 Dataset
ERS	ERMapper .ers Labelled
FIT	FIT Image
RMF	Raster Matrix Format
RST	Idrisi Raster A.1
INGR	Intergraph Raster
GSAG	Golden Software ASCII Grid (.grd)
GSBG	Golden Software Binary Grid (.grd)
R	R Object Data Store
PNM	Portable Pixmap Format (netpbm)
ENVI	ENVI .hdr Labelled
EHdr	ESRI .hdr Labelled
PAux	PCI .aux Labelled
MFF	Vexcel MFF Raster
MFF2	Vexcel MFF2 (HKV) Raster
BT	VTP .bt (Binary Terrain) 1.3 Format
IDA	Image Data and Analysis

Table 4.8.22-8. Valid Formats (2 of 2)

File Type	Description
GTX	NOAA Vertical Datum .GTX
NTv2	NTv2 Datum Grid Shift
USGSDEM	USGS Optional ASCII DEM (and CDED)
ADRG	ARC Digitized Raster Graphics
BLX	Magellan topo (.blx)
Rasterlite	Rasterlite
SAGA	SAGA GIS Binary Grid (.sdat)
KMLSUPEROVERLAY	Kml Super Overlay
XYZ	ASCII Gridded XYZ
HF2	HF2/HFZ heightfield raster

4.8.22.1.3.1 Webservice

The GDAL web service (GdalService) is a web service, build on the same structure as the HegService, which exposes some of the GDAL functionality via an ESI interface. This tool adapter allows requests for GDAL processing to be submitted from Reverb and other clients via the ESI API. Currently, the GdalService only supports reformatting (i.e. conversion from one file format to another).

4.8.22.1.3.1.1 Using the GdalService Interface

GdalService is a web-service that can be invoked via an HTTP GET request. GdalService will usually only be called via the ESI Gateway Service (See EGI below). The URL for GdalService is as follows:

http://host:port/GdalService(_MODE)

followed by your query string (ie appending ?param1=value1&param2=value2...). The following is a table of the parameters available. Note that either a FILE_URLS or FILE_IDS is required but none of the others.

Table 4.8.22-9. GDAL Parameters (1 of 2)

Parameter Name	Values	Description
FILE_URLS	URL String	The data URL identifier for a granule. Only one URL is allowed for processing.
FILE_IDS	Integer ID (unsigned)	The granule-identifier for a granule, only one allowed. GdalService will use this Id, if provided, to attempt to retrieve the file's local access path and the granule's production date for Processing History purposes using SII. If SII is not available, a FILE_URL must be provided. If a FILE_URL and FILE_ID are provided they must correspond to the same granule or an error will be reported and processing stopped.

Table 4.8.22-9. GDAL Parameters (2 of 2)

Parameter Name	Values	Description
OUTPUT_DIR	<Path>	Local path of directory where output file should be placed. This parameter is only used if the client machine and the HegService have access to common storage (e.g. a SAN file system).
FORMAT	<Format>	Format of output file – See list of format options
INCLUDE_META	Yes, No	Directive to include additional meta-information files, including the processing history and granule meta-information files.

Table 4.8.22-10. Additional Parameters For Test or Debugging Purposes

Parameter Name	Value	Description
TIMEOUT	Integer >= 0 (d,h,m,s)	Allows an override to the default configuration setting. Integer can optionally be followed by d (days), h (hours), m (minutes), s (seconds)
DEBUG	Yes, No	Allows an override to the default configuration setting. When YES, all working files are preserved for debugging purposes; when NO all working files are deleted.

4.8.22.1.3.1.2 Examples

Submit a request for GDAL processing of a granule:

http://f4hel01:22500/GdalService_DEV01?FILE_IDS=249600 &FORMAT=GTiff

Notice that special characters (e.g. ' ', etc) are typically escaped using HTTP URL escaping rules. In many cases, this escaping is not strictly required, depending on how your HTTP client handles special characters, but it is recommended to always escape them to avoid confusion or errors. http://www.w3schools.com/tags/ref_urlencode.asp contains a list of rules for escaping special characters.

4.8.22.1.3.1.3 GdalService Interface Configuration File

GdalService uses a properties file, GdalServiceConfig.properties, 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 4.8.22-9 describes the configuration parameters. In some cases, when an alternate name is defined, the name that is defined in the ECS-Assist Configuration settings is provided for reference.

Table 4.8.22-11. GDALService Configuration Parameters (1 of 2)

Parameter Name	Value Description
application.name	Name of this application (Optional).
log.operations.level	Level of logging desired in operational log: NONE, INFORMATION, VERBOSE or XVERBOSE
log.performance.level	Level of logging desired in performance log: NONE, INFORMATION, VERBOSE or XVERBOSE
log.debug.level	Level of logging desired in debug log: NONE, INFORMATION, VERBOSE or XVERBOSE
log.threshold	Size of log files before new ones are created.
log.overwrite	If true, log file will be overwritten for each run.
GdalService.application .workDir	Required for GdalService operations, a directory for working files, sub-directories created within for each request (allows concurrent processing)
GdalService.application .defaultOutDir	Optional, a directory for output files being generated. If undefined, a sub-directory to the current working directory will be created.
HegService.application. copyOutput	Optional (default is false) - true / false. If true, and defaultOutdir is set, forces copy of output files after processing (vs. writing to output-dir during processing), if this is better for performance or other reasons. Note that copy-after-processing is always the case if a given request specifies an output directory.
GdalService.application .LocalGetFileURL (GdalServer_Output_GetFile_URL)	Optional – a URL that will be added to the output file specification if users typically don't have access to file-system of the application itself.
HegService.application. RemoteGetFileURL (GdalServer_Input_GetFile_URL)	Optional – a URL that may be used to retrieve the input file if on a different file-system from the application itself, or if copy of the input file to the working directory before processing is preferred.
GdalService.application .SIIURL (GdalServer_SIIURL)	Required – The URL for the Simple Inventory Interface. Note that the mode for this SII URL should match the mode for the GdalService installation.
GdalService.application .debugFlag	Required – designates if working files are to be kept for debugging purposes. If false, all working files and directories are deleted.
GdalService.application .timeout	Required – GdalService will kill HEG processing if its processing time exceeds this value. Specified as a positive integer, optionally followed by a units designator: d – days, h – hours, m – minutes, s – seconds (seconds is assumed if not specified).
GdalService.application .GdalPath	Required – Location of the GDAL installation on the host where GdalService is running.
GdalService.internal.C onverterScript	Required – Location of the wrapper script which is used to call gdal_translate.
HegService.application. HegPath	Required – Location of the directory where the HEG executables are installed. These executables are used obtain header information about granules for the processing history.
HegService.internal.Co nverterScript	Required – Location of the wrapper script which is used to start the HEG executables.

Table 4.8.22-11. GDALService Configuration Parameters (2 of 2)

Parameter Name	Value Description
REQUEST_CLEANUP_AGE	Required – GdalService spawns a separate thread to periodically check for files in workDir and in defaultOutDir that are older than this specified age. These files and directories are then deleted. Specified as a positive integer, optionally followed by a units designator: d – days, h – hours, m – minutes, s – seconds (seconds is assumed if not specified).
REQUEST_CLEANUP_INTERVAL	Required – For the cleanup thread described above, this values determines the time between clean-up processing runs. Specified as a positive integer, optionally followed by a units designator: d – days, h – hours, m – minutes, s – seconds (seconds is assumed if not specified).
CONTACTNAME	Required – For ESI Standard responses, the contact name
CONTACTEMAIL	Required – For ESI Standard responses, the contact e-mail
ESI.DAAC_NAME	Required – For ESI Standard responses, the DAAC site name
ESI.DAAC_URL	Required – For ESI Standard responses, the DAAC site URL

4.8.22.1.3.1.4 Main Screen

The GdalService is a web service and does not have a main screen.

4.8.22.1.3.2 GDAL Command Line Adapter

GDAL should primarily be referenced using the GdalService described above. However, Release 8.1 also contains a secondary, command line adapter for GDAL reformatting capabilities. This command line adapter is intended to provide a proof of concept for the REST Quick Server (RQS). The RQS allows an ESI compliant tool adapter to be implemented as a command line script. The script can then be configured in the Data Access GUI to utilize the RQS, which will expose a REST style URL under the RQS service that will take in an ESI compliant URL, convert the URL query parameters into command line arguments, and call the configured script. The GDAL command line adapter provides such a command line interface. This adapter is not intended for productioon use, but as an example of how to write a very simple command line adapter.

4.8.22.1.3.2.1 Using the GDAL Command Line Adapter

The GDAL Command Line Adapter accepts three command line parameters. The –FILE_URLS and –FORMAT parameters are both required.

Table 4.8.22-12. GDAL Parameters

Parameter Name	Values	Description
--FILE_URLS	URL String	The local file system path to the input granule. Only one URL is allowed for processing.
--OUTPUT_DIR	<Path>	Local path of directory where output file should be placed. This parameter is only used if the client machine and the HegService have access to common storage (e.g. a SAN file system).
--FORMAT	<Format>	Format of output file – See list of format options

4.8.22.1.3.2.2 Examples

The GDAL Command Line adapter is located in the utilities directory in the x4hel01 host. The script is called 'EcDlDaGdalCmdLineAdapter' and is invoked as follows:

EcDlDaGdalCmdLineAdapter --FILE_URLS <path to input file> --OUTPUT_DIR <desired output directory> --FORMAT <output format>

See the table above for the allowed output formats.

4.8.22.1.3.2.3 Configuration

The GDAL Command Line Adapter configuration file is located at /usr/ecs/<MODE> /CUSTOM/utilities/EcDlDaGdalCmdEnvBash. The values in this configuration file are described in the table below.

Table 4.8.22-13. GDAL Configuration Parameters

Parameter Name	Value Description
outputdir	Default output file directory.
getfileprefix	Optional – path to the local getfile instance, in the case where the output directory is not located on a filesystem that cannot be accessed from the host where EGI and ESIR are running.

4.8.22.1.3.2.4 Main Screen

The GDAL Command Line Adapter is a command line script and does not have a main screen.

4.8.22.1.4 Simple Inventory Interface

The Simple Inventory Interface (SII) provides a simple mechanism to obtain information about granules in the ECS archive, as well as about Services configured in the Data Access system. This interface can be used by applications such as processing tool adapters to obtain necessary information about granules being processed (such as granule file location), as well as by operators who would like to get granule information without having to write complex SQL queries. Output is provided in a number of formats, including HTML and XML.

Simple Inventory Interface API:

The Simple Inventory Interface generates many types of output in multiple formats. The API for XML output of granule metadata is documented in an XML schema located at http://newsroom.gsfc.nasa.gov/esi/8.1/schemas/SimpleInventoryInterface.xsd and in a more user friendly HTML format at http://newsroom.gsfc.nasa.gov/esi/8.1/schemas/doc/sii/index.html .

4.8.22.1.4.1 SII Webservice

4.8.22.1.4.1.1 Using the SII Webservice

The Simple Inventory Interface (SII) is a web service called via the HTTP GET method. The SII will provide information about the requested granule or service in the ECS system. The URLs for the SII service are as follows:

To obtain basic inventory information about a granule you can use the following URL:

http://<host>:<port>/ SimpleInventoryInterface (_MODE)/granule/{granuleId}.{extension}

Where {granuleId} is the ECS GranuleId of the granule in question and {extension} indicates the desired output format of the response. Valid values for {extension} are: xml, txt, json, html.

To obtain HDF file information about a granule, you can use the following URL:

http://<host>:<port>/ SimpleInventoryInterface (_MODE)/geofile/{granuleId}.{extension}

Where {granuleId} is the ECS GranuleId of the granule in question, and {extension} is xml. Currently only an xml response is supported.

To obtain a list of granules with a given LocalGranuleId, you can use the following URL:

http://<host>:<port>/ SimpleInventoryInterface (_MODE)/granule.html?lgid={localGranuleId}

Where {localGranuleId} is a local granuleId string.

To obtain information about a configured ESI processing service, you can use the following URL:

http://<host>:<port>/ SimpleInventoryInterface (_MODE)/service/{serviceName}.{extension}

Where {serviceName} is the name of teh ESI service in question and {extension} indicates the desired output format of the response. Valid values for {extension} are: xml, txt, json.

4.8.22.1.4.1.2 SII Webservice Examples

Get inventory metadata for granule 291308, in HTML format:

http://f4hel01:22500/SimpleInventoryInterface_DEV01/granule/291308.html

Get HDF header and structure information for granule 291308, in XML format:

http://f4hel01:22500/SimpleInventoryInterface_DEV01/geofile/291308.xml

Get an HTML page containing a list of granules with the LocalGranuleId "AST_L1B_00302192011160332_20110221095610_9305.hdf"

http://f4hel01:22500/SimpleInventoryInterface_DEV01/granule.html?lgid=AST_L1B_00302192011160332_20110221095610_9305.hdf

Get metadata for the ESI processing service named 'HEG', in JSON format:

http://f4hel01:22500/SimpleInventoryInterface_DEV01/service/HEG.json

4.8.22.1.4.1.3 Configuration

Simple Inventory Interface configuration files are generated by the SII MKCFG utility. This utility should be run through the ECS Assist utility after installing SII. This MKCFG will generate two configuration files, the contents of which are described below.

Table 4.8.22-14. EcDlDaSii.properties

Parameter Name	Value Description
SII.RemoteGetFileURL	URL of the ESI getfile web service. This is used to obtain granule data files if SII is running on a machine that does not have access to the Datapool filesystem. If SII is running on a Datapool filesystem connected host, then this value should be blank.
SII.GeoFileInfo.WorkingDir	Working directory to store working files in. Must be writeable by user tomcat.
HegService.internal.ConverterScript	Location of the HEG wrapper script which will be used to invoke the HEG header extractor.
GdalService.application.GdalPath	Path to gdal bin directory

Table 4.8.22-15. web.xml

Parameter Name	Value Description
dbUsername	Database username
dbIp	Sybase server host name
dbPort	Sybase server port
dbName	Name of database

4.8.22.1.4.2 SII Command Line Client

The SII command line client offers some of the functionality of the SII web service via a command line utility. Behind the scenes, this client actually calls the web service, but this utility allows a user to obtain granule metadata without using a web browser or other HTTP client. The SII command line client only provides the general granule metadata functionality of SII (and not the service metadata or granule header information).

4.8.22.1.4.2.1 Usage

The command line utility is located in the mode utilities directory and is called from teh command line as follows:

EcDaSiiCmdStart <mode> <host> <port> <additional_params>

Where <mode> is the current ECS mode, <host> and <port> are the host and port where the SII web service is running. And where <additional_params> is composed of the following:

-ecsid <arg> (**Required**) GranuleId of the granule of interest.
-h Print help message.
-help Print help message.
-outdir <arg> (Optional) Full path of the output file. Default To current directory './'.
-outfile <arg> (Optional) Name of the output file. Default to 'sii_output'.<outFormatOpt>.
-outformat <arg> (Optional) Specify format of the outputs: xml, stdout, txt, json. Default to standard output (stdout), which implies txt formatted output.

<mode>, <host>, <port> and the additional parameter ***ecsid*** are required. All the rest are optional.

4.8.22.1.4.2.2 Examples

This command retrieves the information of granule ***216778*** with no optional arguments:

> EcDaSiiCmdStart DEV01 f4hel01 22500 –ecsid 216778
File format: (default value) ***stdout***
File name: not available, when format is stdout.
Output file directory: not available, when format is stdout.

This command specifies the output file format:

> EcDaSiiCmdStart DEV01 f4hel01 22500 –ecsid 216778 –outformat xml
File format: ***xml***
File name: (default file name) ***sii_output.xml***
Output file directory: (default value) current directory

This command specifies output file directory and format:

> EcDaSiiCmdStart DEV01 f4hel01 22500 –ecsid 216778 –outformat html –outdir /home/bsun/tmp
File format: ***html***
File name: ***sii_output.html***
Output file directory: ***/home/bsun/tmp***

This command specifies output file name and directory and its format:

> EcDaSiiCmdStart DEV01 f4hel01 22500 –ecsid 216778 –outformat xml –outfile 216778_output –outdir /home/bsun/tmp
File format is set to: ***xml***
Fle name: ***216778_output.xml***
Output file directory: ***/home/bsun/tmp***

4.8.22.1.4.2.3 Configuration

The SII command line client does not have any configuration other than the parameters provided on the command line and the configuration provided for the back end SII web service.

4.8.22.1.5 REST Quick Server (RQS)

The REST Quick Server (RQS) is used to convert a command line tool adapter into a web service tool adapter. The RQS accomplishes this by acting as the intermediate layer. RQS will act as the web service then call the command line tool adapter located on the same host. Then it will return the contents of the command line tool adapter as its response.

4.8.22.1.5.1 Using the REST Quick Server (RQS)

The REST Quick Server (RQS) is a web service called via the HTTP GET method. The RQS will translate the request parameters into command line parameters. Although RQS can be called manually, it will usually only be called via ESI. The URL for RQS is as follows:

http://host:port/rqs(_MODE)/{service}

where "{service}" is the name of the Command Line Tool Adapter Service configured in the GUI. Request parameters can be appended via the normal HTTP GET method (ie. ?param1=value,param2=value,…)

4.8.22.1.5.2 RQS Example

Calling the GDAL Command line adapter through RQS to convert format to JPEG:

http://i4oml01:45443/rqs_DEV01/GDAL?DATASET_ID=ASTER+Level+1B+Data+Set+Registered+Radiance+at+the+Sensor+ONEV005&FILE_IDS=220253&FORMAT=JPG&FILE_URLS=/datapool/DEV01/user/FS2/MOST/MOD10A1.005/2008.02.01/MOD10A1.A2008032.h30v09.005.2008034170146.hdf1251826105.46239.RGEN.hdf

4.8.22.1.5.3 RQS Configuration

The REST Quick Server uses a properties file, EcDlDaRqs.properties, 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 1.1.1.1,5.3-2 describes the configuration parameters.

Table 4.8.22-16. Configuration Parameters

Parameter Name	Value Description
application.name	Name of this application.
program.id	ID 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.
sii.url	The URL to connect to the Simple Inventory Interface.

RQS gets its service information from the Simple Inventory Interface (See SII section).

4.8.22.1.5.3.1 Configuring a Command Line Tool Adapter via RQS

The RQS is only able to expose command line tool adapters which are configured via the Data Access GUI (See the Data Access GUI section of this document). First, ensure that the command line adapter being exposed accepts arguments in the form of "--<param name> <param value>" where <param_name> is one of the ESI request parameters defined at http://newsroom.gsfc.nasa.gov/esi/8.1/schemas/doc/req/eesi/eesi_request-Type.html. The adapter does not have to do anything meaningful with each parameter, but must be able to accept all of them without failure. The command will receive at a minimum, the FILE_IDS and FILE_URLs parameters, and should use one of them to obtain its input files. The configuration of the command in the Data Access GUI should be done in such a way that only those options which the command will accept and use in its processing are configured (e.g. if the tool does not support reprojection, do not specify any valid projections for that service).

In order to configure the command line adapter in the Data Access GUI, follow the instructions in the Data Access GUI section to create a new service. In the service configuration form, specify the <host>:<port> under which RQS is running in the "Host" field, and the path to the command line adapter executable in the "URL or Service Path" field.

An example command line adapter is provided with Release 8.1. More information about the GDAL Command Line Adapter can be found in the GDAL section. This adapter is by default configured as the "GDAL_CMD" service in the Data Access GUI. This service is not by default

linked to any collections, but is provided simply as an example. Below is a screen shot of the configuration page for that service. Note the value "f4hel01:22500" in the 'Host' field and "/usr/ecs/DEV01/CUSTOM/utilities/EcDlDaGdalCmdLineAdapter" in the "URL or Service Path" field.

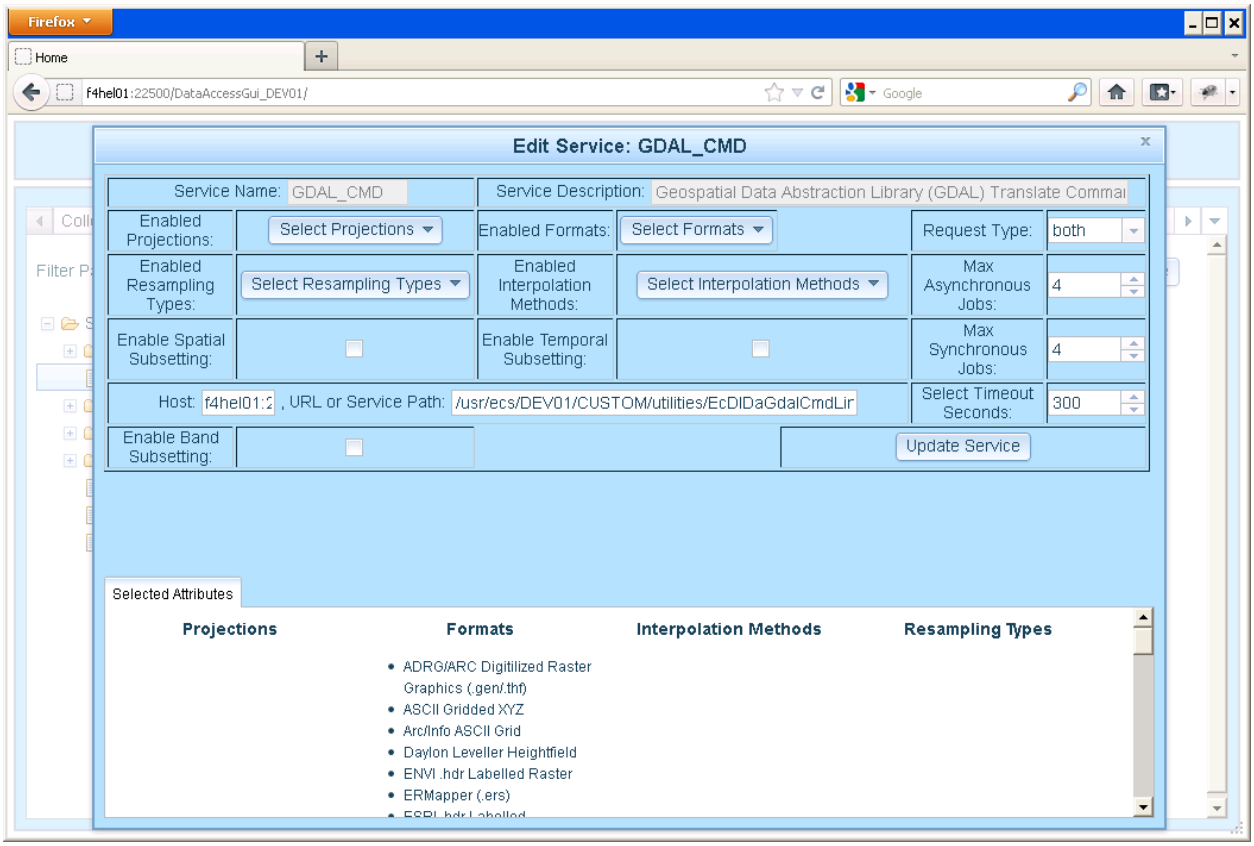


Figure 4.8.22-10. Data Access GUI Edit Service Page for the GDAL_CMD Tool Adapter

4.8.22.1.5.4 RQS Main Screen

The REST Quick Server does not have a main screen. It is a web service. Please see RQS Usage above.

4.8.22.1.6 Data Access GUI

The Data Access GUI (DA-GUI) is used to monitor and configure the ESI standard applications (services), and configure their access to the various ECS Data Collections.

4.8.22.1.6.1 Using the Data Access GUI

The DA-GUI is a web application started in a Web browser via an HTTP address (URL). The URL for the DA-GUI is as follows:

http://host:port/DataAccessGui(_MODE)

The GUI is designed to be run using the Firefox web browser. Using any other browser will result in a warning message and potentially unexpected behavior.

4.8.22.1.6.1.1 Login and Logout

By default, upon initial installation, the GUI is not secured. This is because there are some initialization scripts that must access the GUI webservice and are not able to login. Also, some DAACs will choose to leave the GUI open. In order to support the open configuration, the configuration parameter SECURED_URL_PATTERN is initially set to '/none'. After the initialization steps have been completed, if the DAAC wishes to enabled authentication, this parameter should be set to '/' in the ECS Assist GUI, and the GUI MKCFG must be run again. After this, and any other configuration changes, the web service must be manually restarted using the tomcat manager.

When SECURED_URL_PATTERN is set to '/', the Data Access GUI is secured against open access to its web pages and web services. All pages and resources are secured using J2EE Form-based authentication and authorization, in this case administered by the Tomcat web-application server. The Data Access GUI Application registers all pages and services to have constrained access, requiring a specific user-role for access. The Tomcat installation is then configured to accept certain user/password combinations and to grant those users selected user-roles. See your system administrator, if necessary, for information about being granted access to the Data Access GUI.

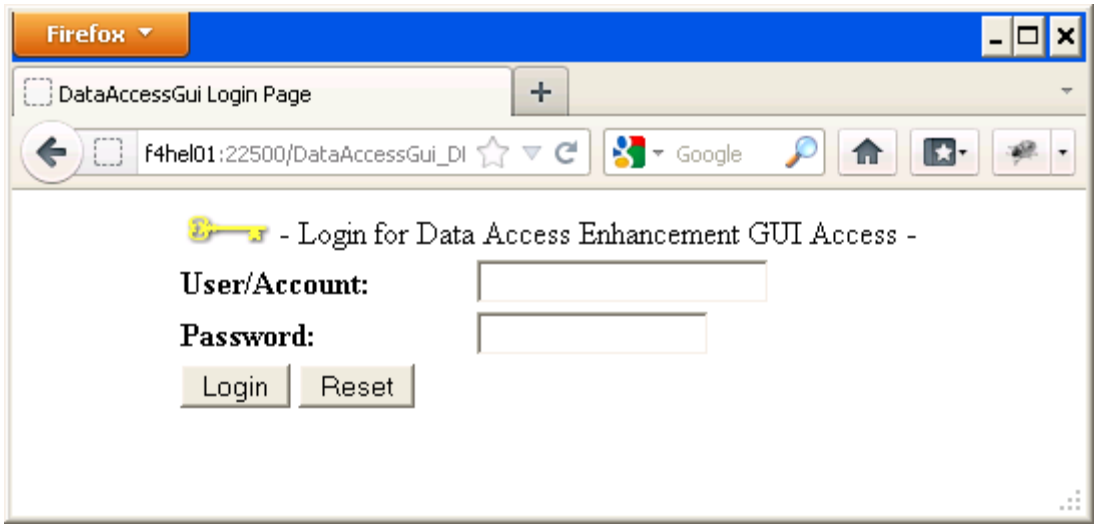


Figure 4.8.22-11. Data Access GUI Login Page

An invalid login attempt is simply referred to an error page and the user is directed to try again (the error pages and login/logout are open to all users). A valid login attempt using an account without the proper role association is directed to a different error page and will force a logout if attempting to return to the login screen.

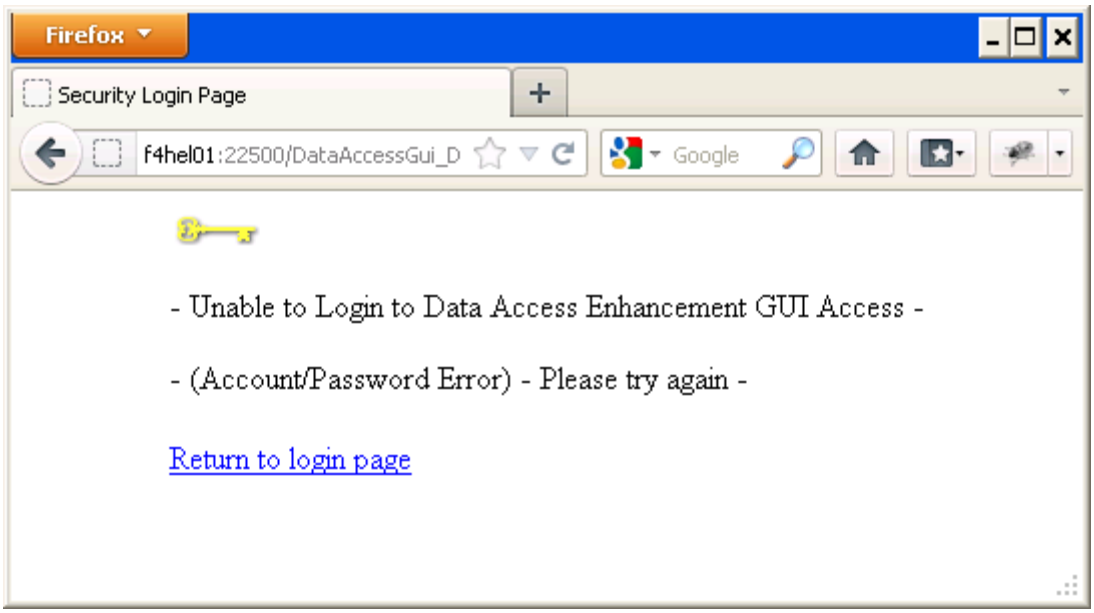


Figure 4.8.22-12. Data Access GUI Invalid Login Attempt Page

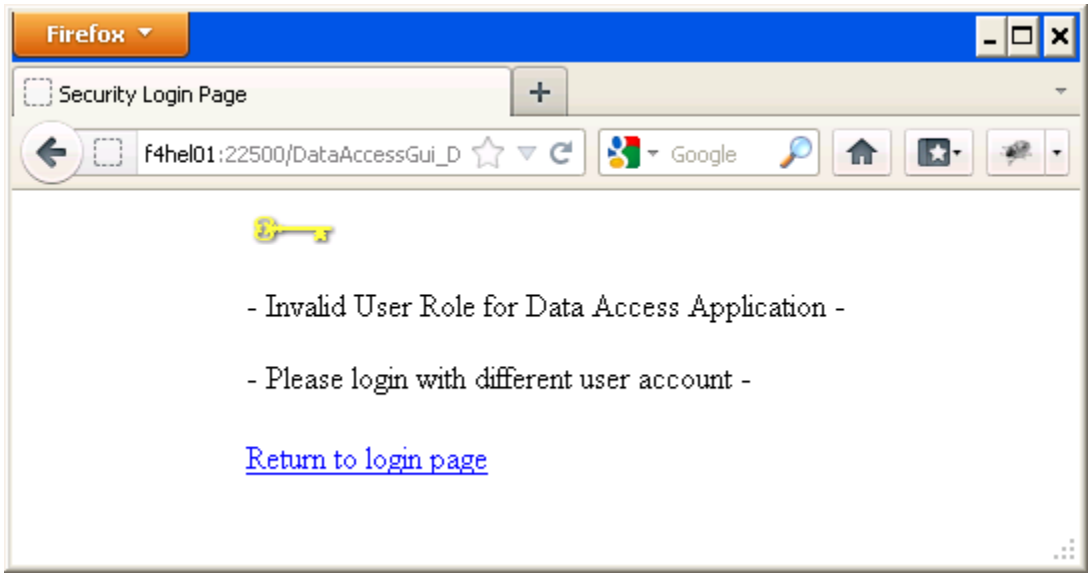


Figure 4.8.22-13. Data Access GUI Login with Incorrect User Role Page

A logout tab is offered on the main application page that will allow the user to explicitly logout of the user account used for access.

4.8.22.1.6.1.2 The Collection and Services Tabs

In the DA-GUI, a service entry provides the registration mechanism for connecting EGI & ESI with a set of applications (services) which perform data processing. Creating a service entry is the first step to establishing a new or custom application for the system. The DA-GUI comes preconfigured for two services: HegService and GdalService. The Service Tab provides access to services and the collections that are configured (mapped) for that service.

The Collection tab provides access to collections, and the services that are configured (mapped) for that collection. Configuring a service to a collection (and vice-versa) enables the service on that collection. The service will now show up for a collection as an available service. Note that the service entry in the collections tab and the collection entry in the services tab refer to the same service-collection mapping entry. Upon initial installation of Data Access, the HEG and GDAL services will be automatically enabled on all collections where AmCollection.ConvertEnabledFlag = 'Y'. Additional collections can be manually configured in the GUI.

In addition to mapping a service to a collection, a given mapping may be enabled or disabled. This is a convenience for enabling/disabling the mapping without actually deleting the mapping. Further, the options associated with configuring a service may be tailored specifically to a collection-service mapping. See the section below for a discussion of the options for configuring a service-collection mapping.

For the collections tab, a "tree" listing starts with the collections at the top level and the associated services at the second level – per collection. Note that the tree labels are provided from the short names of the collection and service respectively, while the full description is available as a hover-over call-out box (tool-tip).

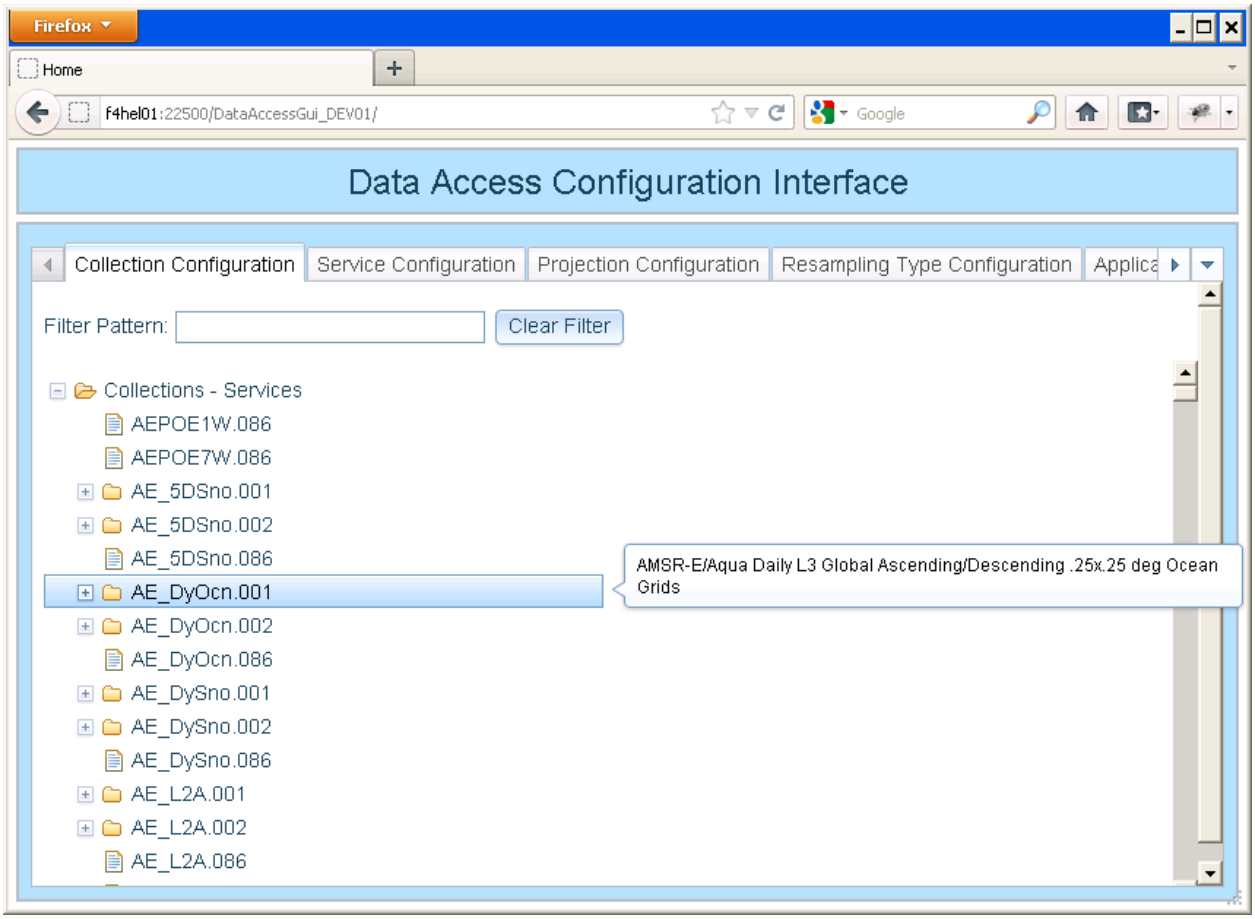


Figure 4.8.22-14. Data Access GUI Collection Configuration Page

A Filter Pattern feature appears at the top of the form. Entering text in this field will immediately start to exclude collections from the list below that do not match the text entered. The filter assumes a wildcard before and after the entered text, and the text-matching is not case sensitive. The clear-filter button will restore the full list of collections. Note that the filter will match against both the short name and long name of a collection.

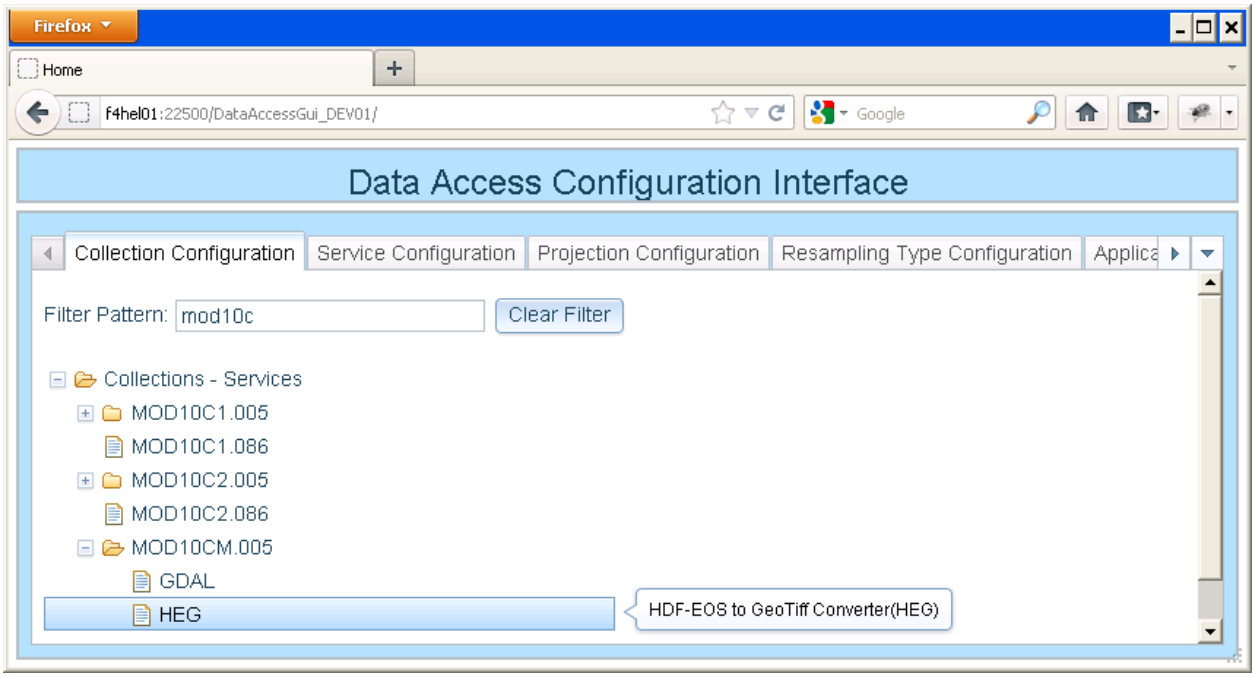


Figure 4.8.22-15. Data Access GUI Collection Configuration Filter Example

A right click on the collections item shows a menu of commands, to:

(a) configure a new service for the collection,
(b) enable all configured (mapped) services,
(c) disable all mapped services,
(d) map all available services to the collection,
(e) unmap all services currently mapped to the collection,
(f) use this collection as a model to set the service options for another collection, or
(g) copy the service options of another (already chosen) collection to this collection.

A right click on a service entry (under a collection) provides a menu of commands to:

(a) remove the service from the collection, or
(b) to enable/disable just that service-collection mapping.

A double-click on the service listing will bring up the Configure-Service-for-Collection dialog. (There is no function associated with double-clicking a collection item).

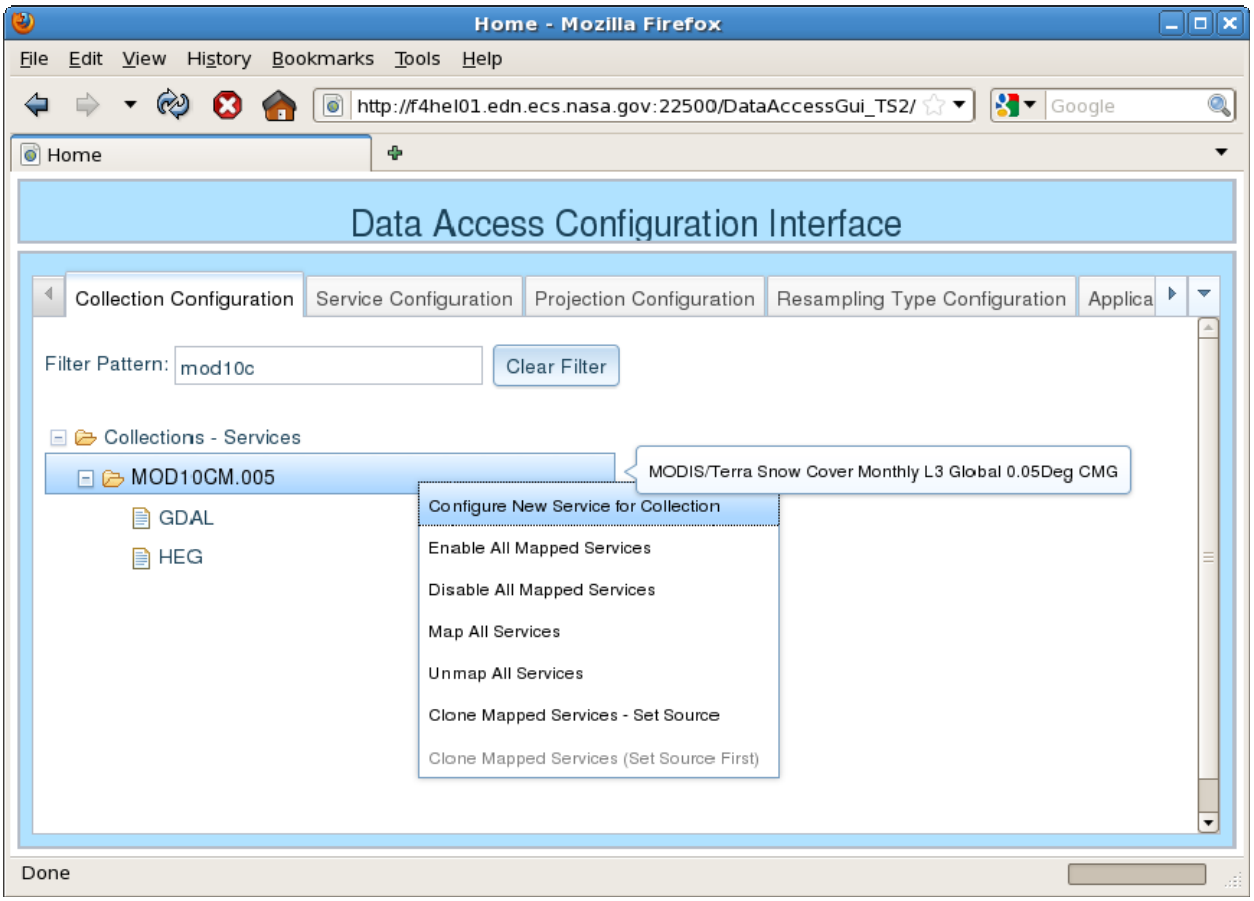


Figure 4.8.22-16. Collection Context Menu

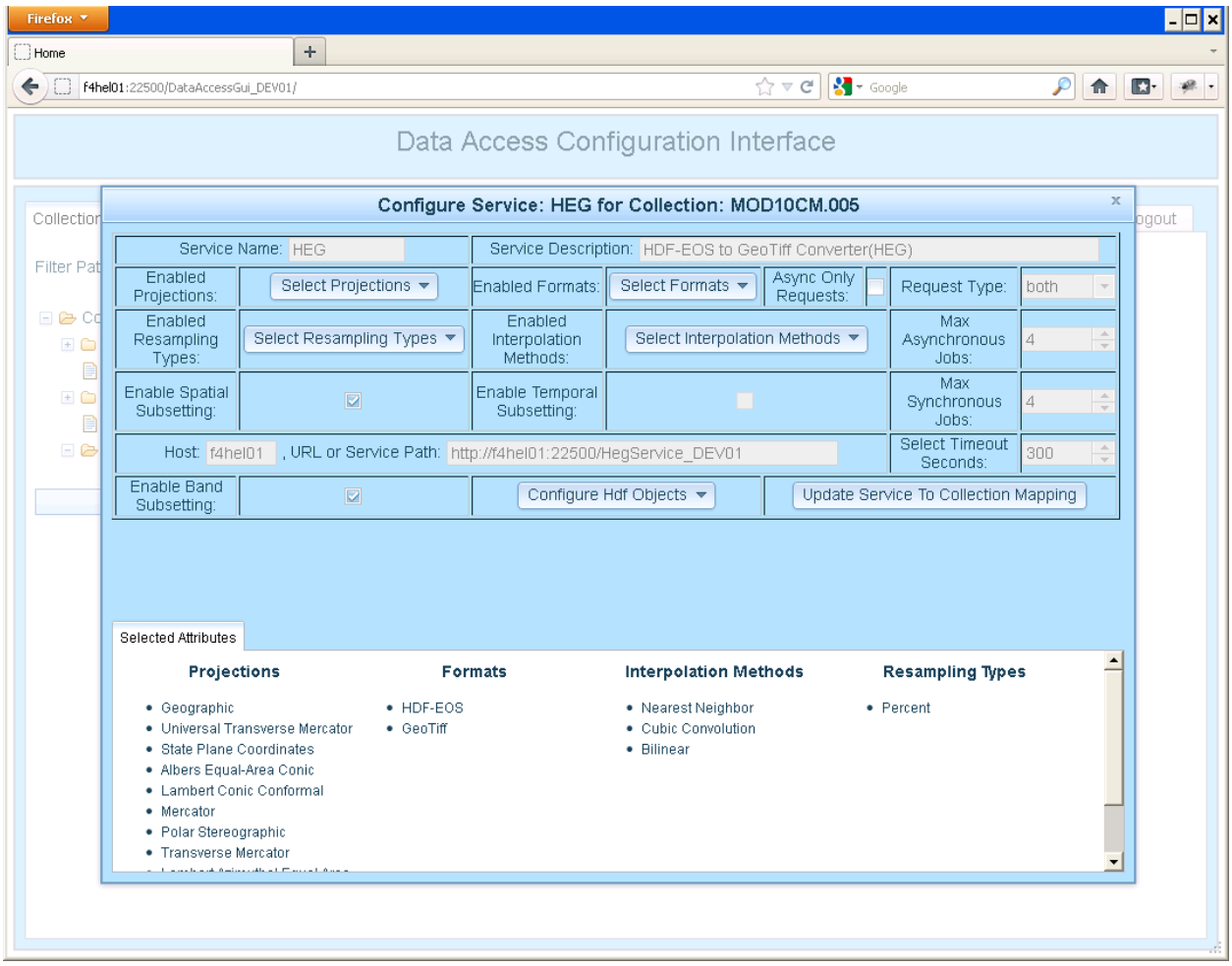


Figure 4.8.22-17. Data Access GUI Collection Configuration configure Service for Collection Page

For the services tab, a “tree” listing starts with the service at the top level and the associated collections at the second level – per service. A Filter Pattern feature appears at the top of the form. As for the Collections tab, entering text in this field will immediately start to exclude services from the list below that do not match the text entered. An Add-New-Service button is available on the right allowing a user to create new service entries.

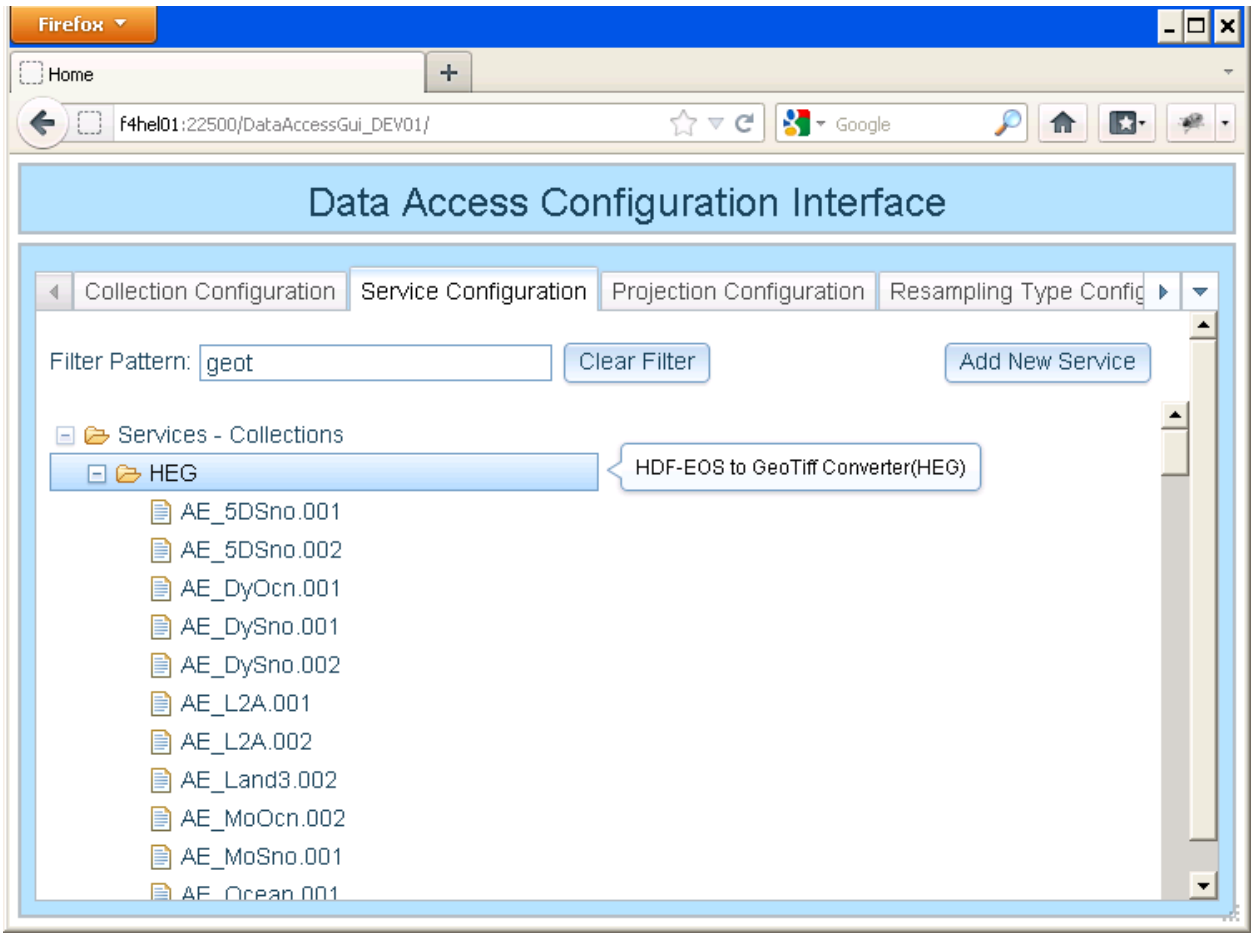


Figure 4.8.22-18. Data Access GUI Service Configuration Page

A right click on the service item shows a menu of commands to:

(a) enable or disable all mapped collections,
(b) to make a copy of a service entry (clone),
(c) delete a service entry,
(d) map all collections to this service,
(e) unmap all collections currently mapped to this service,
(f) use this service as a model to populate options for another service, or
(g) populate this service with the options from another (already selected) service.

Double-clicking a service item brings up the Edit Service Configuration dialog.

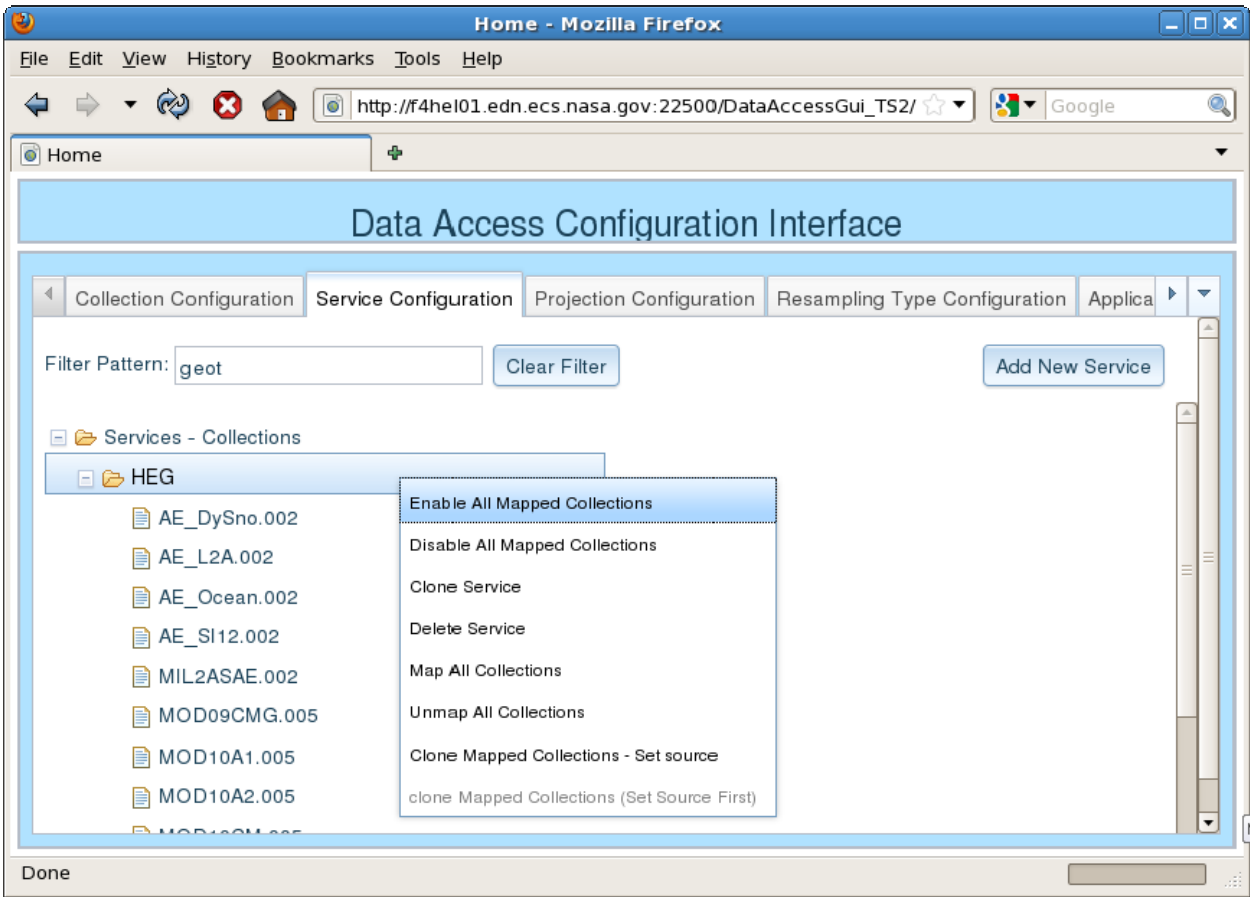


Figure 4.8.22-19. Service Context Menu

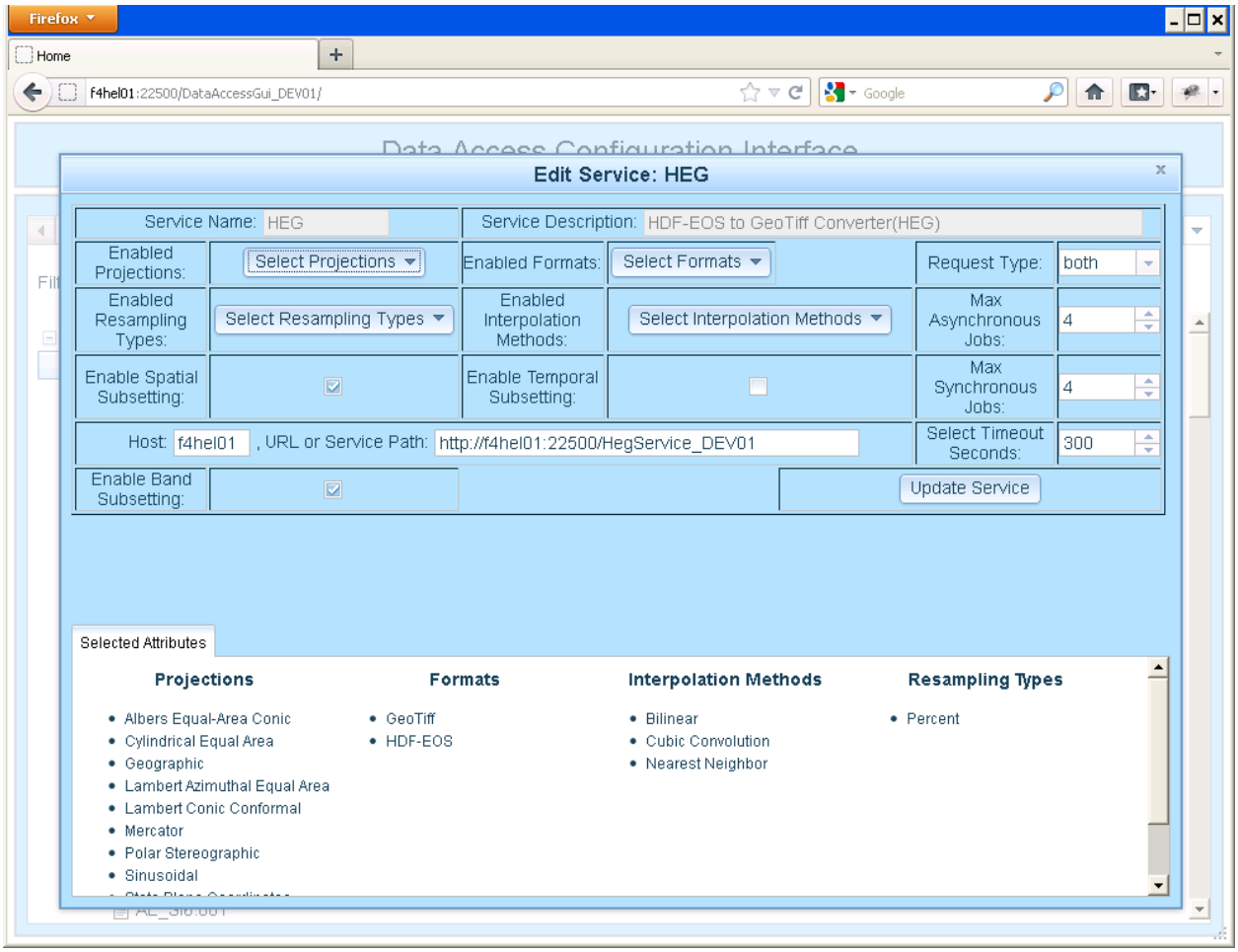


Figure 4.8.22-20. Data Access GUI Service Configuration Edit Service Page

A right click on the collection item (under a service item) shows a menu of commands to:

(a) remove the collection from the service or
(b) to enable/disable that service-collection mapping.

Double-clicking a collection under service item brings up the Edit Service for Collection Configuration dialog. This is the same dialog as for the Edit Service Configuration dialog, but allows editing the service configuration specifically for the mapped collection item. See below for a description of configuring a Service-Collection mapping.

4.8.22.1.6.1.3 Configuring a Service

The Service tab at the top of the main DA-GUI application web-page provides access to the service configuration capabilities. The Add New Service button on the right allows a new service to be created. Clicking on this button pops up the Configure-Service dialog. (Note that this same dialog is used for new services, editing existing services and editing the configuration for a Collection-Service mapping. The title identifies what mode the dialog is in). Double-

Clicking on the service name in the tree listing allows editing the service configuration and again pops-up the Configure-Service dialog.

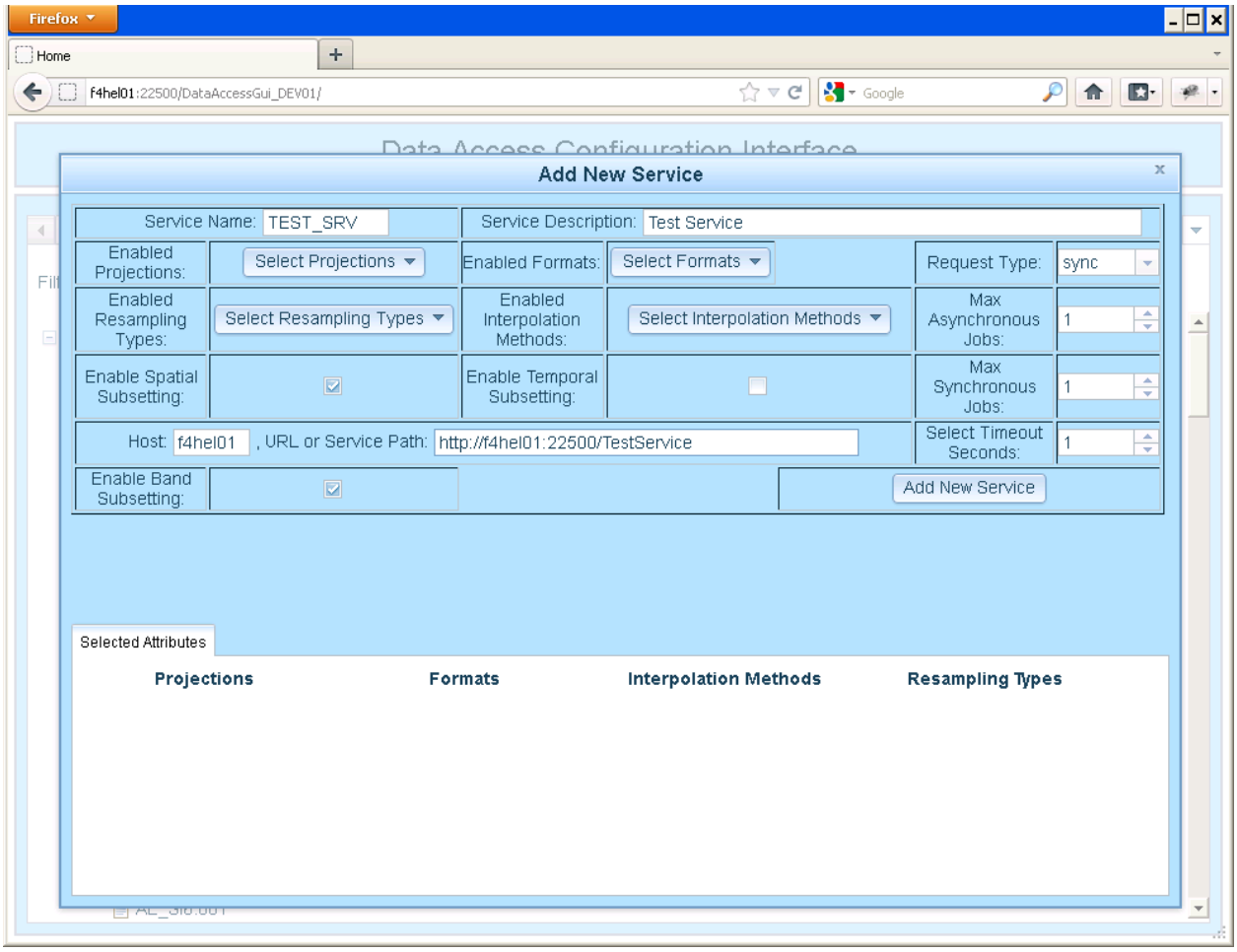


Figure 4.8.22-21. Data Access GUI Service Configuration Add New Service Page

The main fields for a Service are the name, description, host and URL path. Once created, the name and description cannot be altered. Additional fields allow the user to specify the Request-Type (Synchronous, Asynchronous or both), the Max-Asynchronous-Jobs, the Max-Synchronous-Jobs and a job timeout value in seconds. Check-boxes allow the user to select or deselect Spatial Sub-setting, Temporal Sub-setting and/or Band Sub-setting (data object, field and or band sub-setting).

The display area at the bottom of the dialog shows the associated projections, formats, Interpolation-Methods and resampling-types for the service. Buttons above allow access to the drop-down panels for adding and removing items from the service configuration.

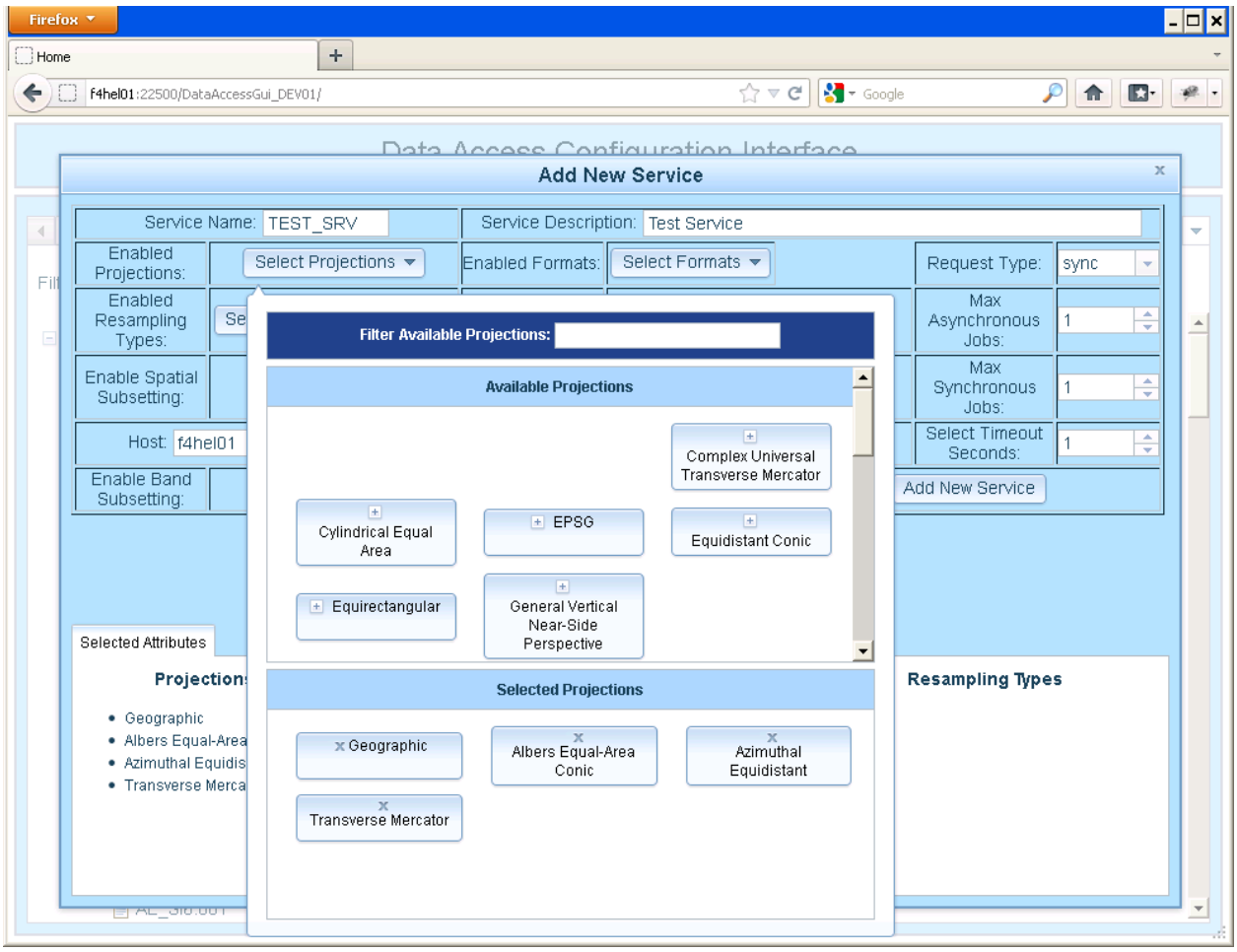


Figure 4.8.22-22. Data Access GUI Service Configuration Filter Available Projections Page

In addition to creating a service with the Add New Service button, double-clicking an existing service allows the user to edit an existing service, using the same Service Configuration dialog panel.

Note that the Update-Service or Add-Service button must be pressed to save any changes made on this dialog.

4.8.22.1.6.1.4 Configuring a Service-Collection Mapping

In addition to configuring a service in general, the system allows configuring a service specifically for a given service-collection mapping. While the dialogs look nearly the same, the changes made while editing a service-collection mapping will affect only that service-collection mapping, and not any other service-collection mapping nor the general configuration for the service. In general, only options which have been configured for the service are available for the

service to collection mapping. For example, if a projection has not been configured in the service configuration, it will not be available for the service to collection mapping.

One feature is available on the Edit Service for Collection dialog that is not available when editing the general configuration for a service. This is the Configure HDF Objects button. Clicking on this button shows a drop-down panel for enabling/disabling the various HDF objects/fields/bands for the collection. In general, all the available objects/fields/bands for a collection are available to a given service. This feature, however, allows the user to custom configure a given service to access only selected bands. For example, many collections contain bands that cannot be processed by e.g., the HegService. These bands can be individually disabled in the Configure Service for Collection dialog. (Note, however, the disabling of non-heggable bands is typically done automatically for all collections configured by the initial setup script and does not require manual editing).

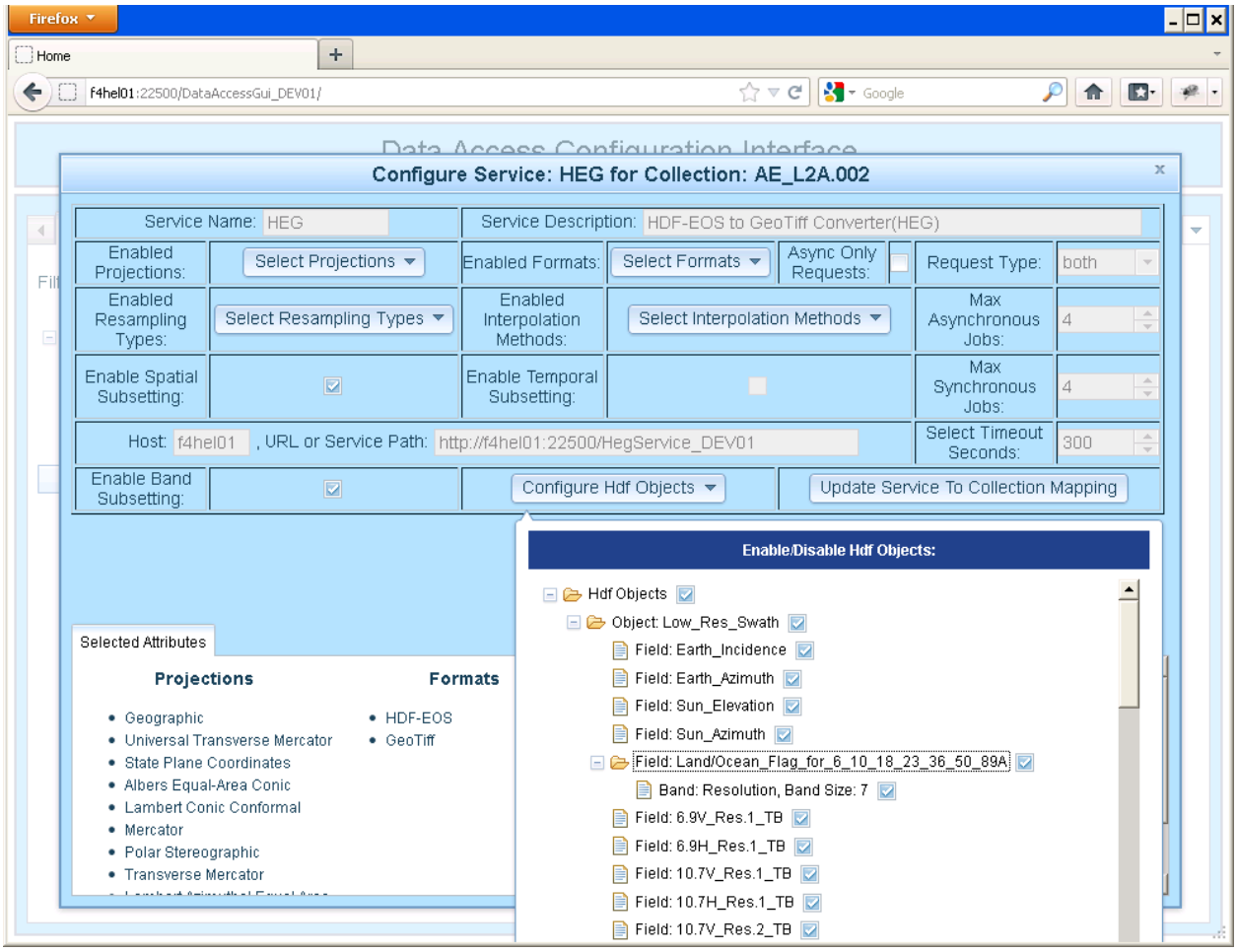


Figure 4.8.22-23. Data Access GUI Service Configuration Configure HDF Objects Page

Note that the Update-Service-To-Collection-Mapping button must be pressed to save any changes made on this dialog.

4.8.22.1.6.1.5 Configuring Projection Parameters

Within the ESI standard, a request may specify a specific output projection and a set of projection parameters. The Projection Configuration tab of the Data Access GUI allows the user to configure a specific set of projection parameters for a given projection. The Projection Configuration tab comes pre-configured with a list of common projection types. (Any additions currently require a manual database update). A Filter Pattern feature appears at the top of the form. As for the Collections tab, entering text in this field will immediately start to exclude Projections from the list below that do not match the text entered. Clicking on the Clear-Filter button will restore the full list of projections.

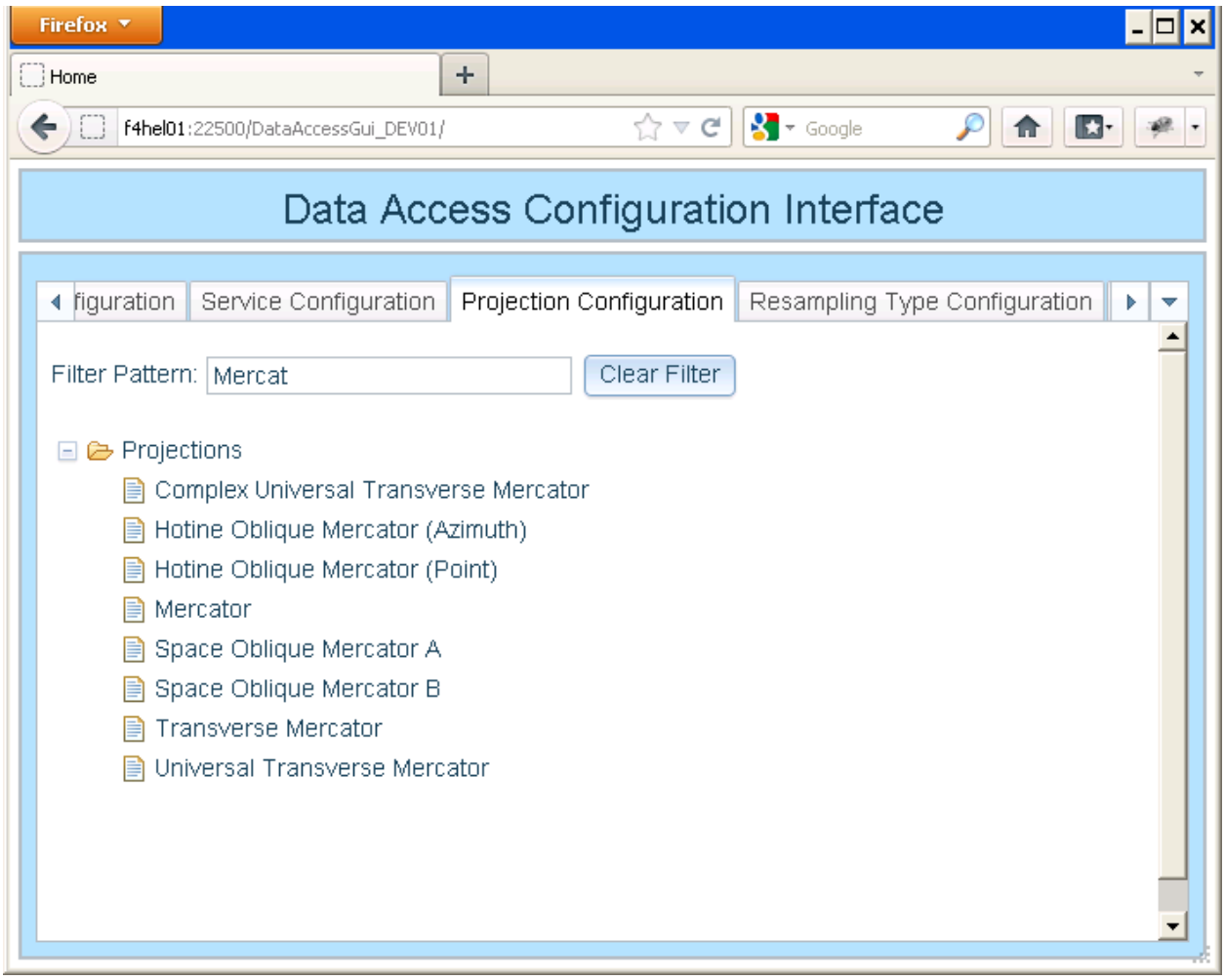


Figure 4.8.22-24. Data Access GUI Projection Configuration Page

Double-Clicking on a projection item pops-up the Configure Projection Parameters dialog. This dialog shows on the left side a list of parameters, and on the right, several fields defined for that projection parameter. The fields include a description, the name, the units and data-type (integer, string), a default value and a required check-box (yes/no).

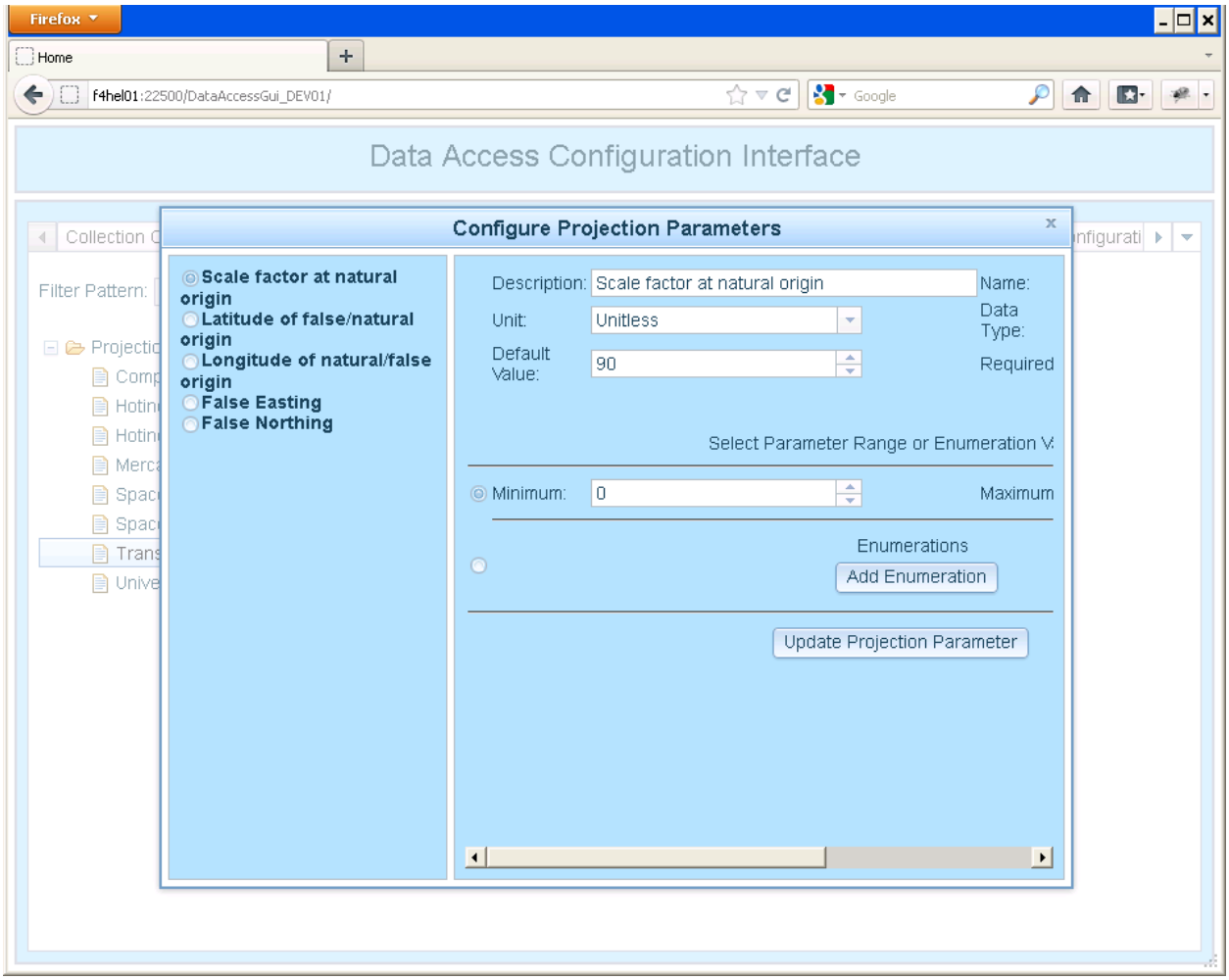


Figure 4.8.22-25. Data Access GUI Projection Configuration Configure Parameters Page

In addition, the user has the choice of setting a range of values (min-max) for the item or an enumerated list of legal values. The radio buttons for these two choices automatically disable the other choice. Selecting Enumerations enables the user to add enumeration values, while any existing enumeration values can be edited or deleted.

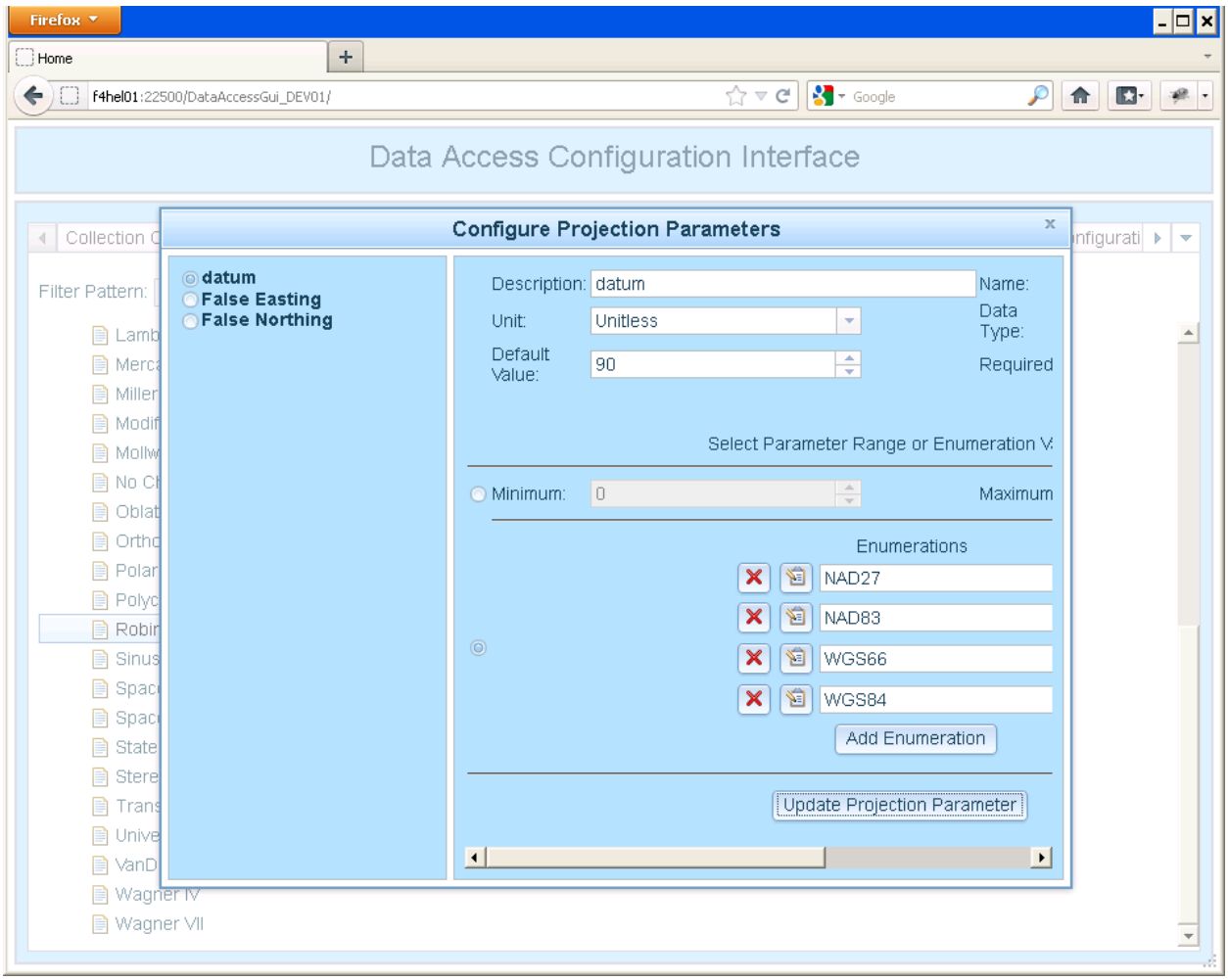


Figure 4.8.22-26. Data Access GUI Projection Configuration Configure Parameter Enumerations Example

Note that the Update-Projection-Parameter button must be pressed to save any changes made on this dialog.

4.8.22.1.6.1.6 Configuring Resampling Types (Dimensions)

Within the ESI standard, a request may specify output resampling and the pixel-size to use. The Resampling Type Configuration tab of the Data Access GUI allows the user to configure the resampling types (units/dimensions) for specifying pixel-size. The Resampling Type Configuration tab comes pre-configured with a list of common resampling types. (Any additions currently require a manual database update). A Filter Pattern feature appears at the top of the form. As with the Collections tab, entering text in this field will immediately start to exclude resampling-types from the list below that do not match the text entered. Clicking on the Clear-Filter button will restore the full list of resampling-types.

Double-Clicking on a Resampling-Type item pops-up the Configure Resampling Type dialog. This dialog shows several fields defined for resampling types, including a description, the name, the units and data-type (integer, string).

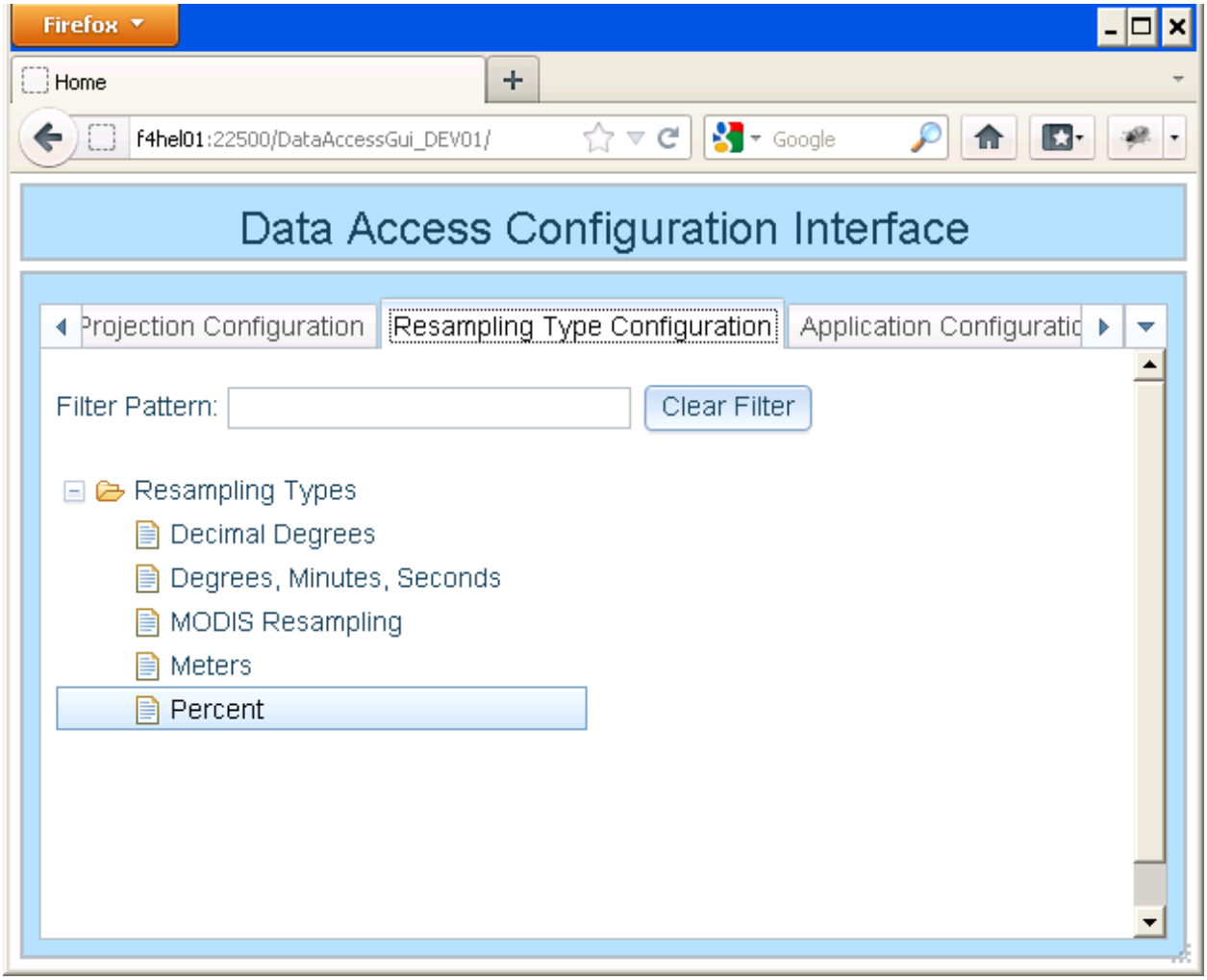
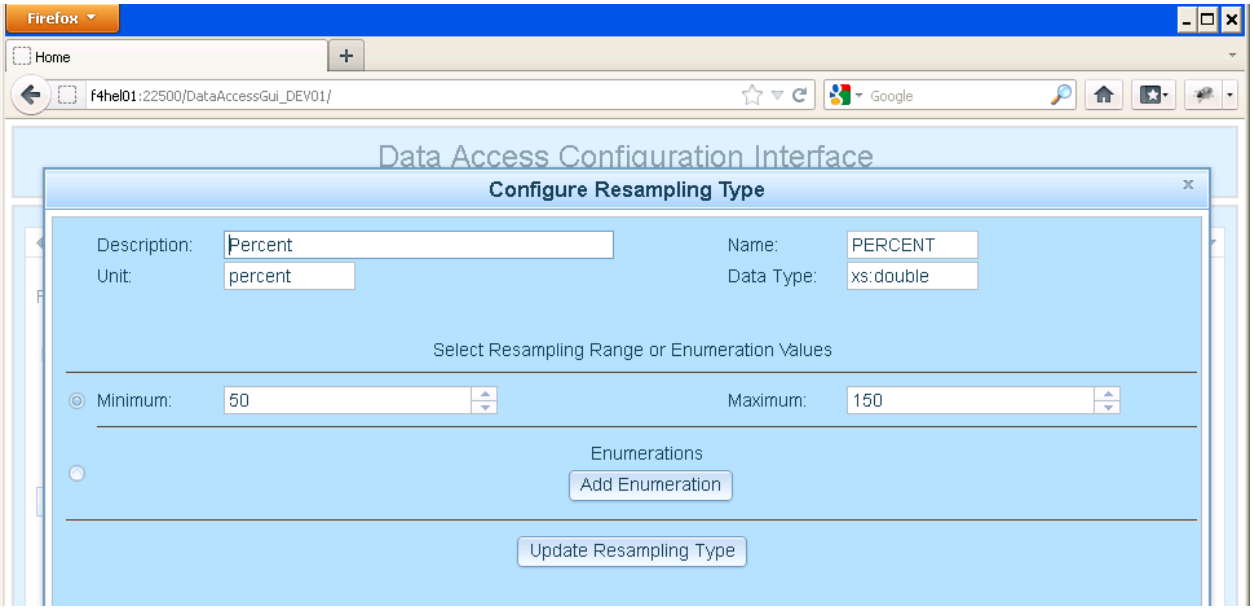


Figure 4.8.22-27. Data Access GUI Resample Type Configuration Page

In addition, the user has the choice of setting a range of values (min-max) for the item or an enumerated list of legal values. The radio buttons for these two choices automatically disable the other choice. Selecting Enumerations enables the user to add enumeration values, while any existing enumeration values can be edited or deleted.



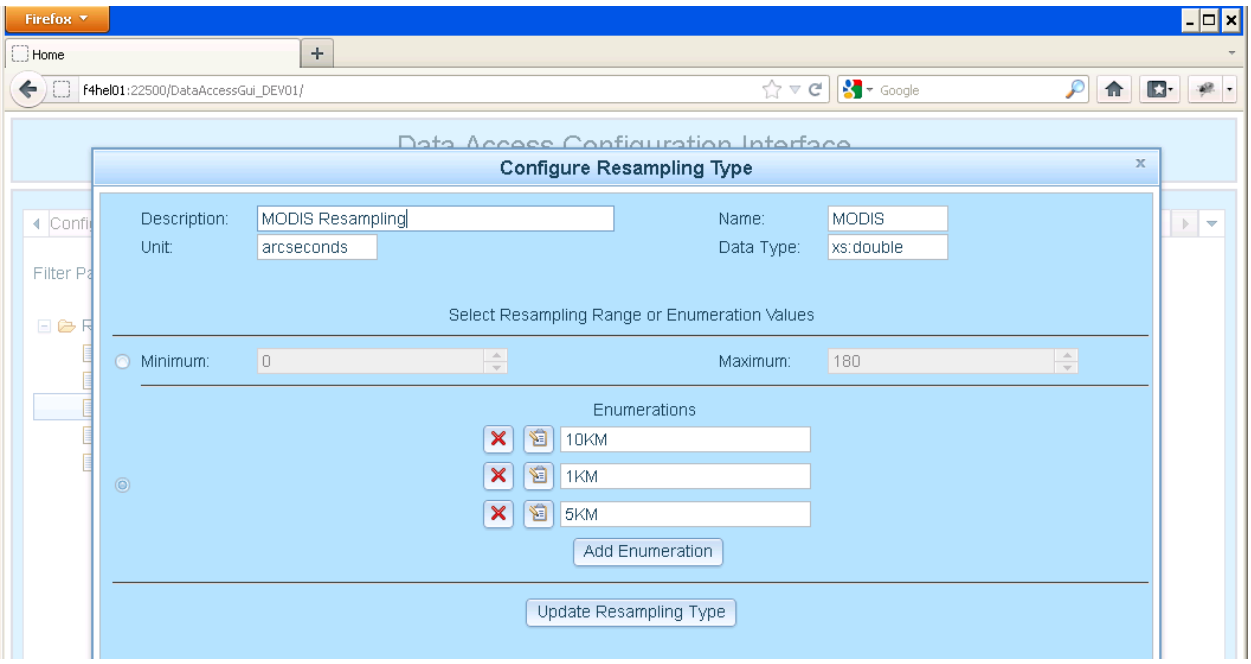


Figure4.8.22-28. Data Access GUI Resample Type Configuration Configure Resampling Type Page

Note that the Update-Resampling-Type button must be pressed to save any changes made on this dialog.

4.8.22.1.6.1.7 Configuring Application Parameters

The Application Configuration tab of the Data Access GUI presents the user with a list of application configuration parameters and parameter groups. These parameters are not specific to a specific application, but may be grouped that way. Each parameter group contains a number of parameters. A parameter may be double-clicked for editing, while two buttons on the right of the page allow for adding and deleting configuration parameters. A Filter Pattern feature appears at the top of the form. As with the Collections tab, entering text in this field will immediately start to exclude Parameter-Groups from the list below that do not match the text entered. Clicking on the Clear-Filter button will restore the full list of Parameter-Groups.

A grouping occurs when a parameter is edited and defines a group-name. If the group-name has been used before, the parameter is added to that group. If the group-name has not been used before, the group is created in the listing. If the last parameter containing a given group-name is deleted, then that group disappears along with the parameter.

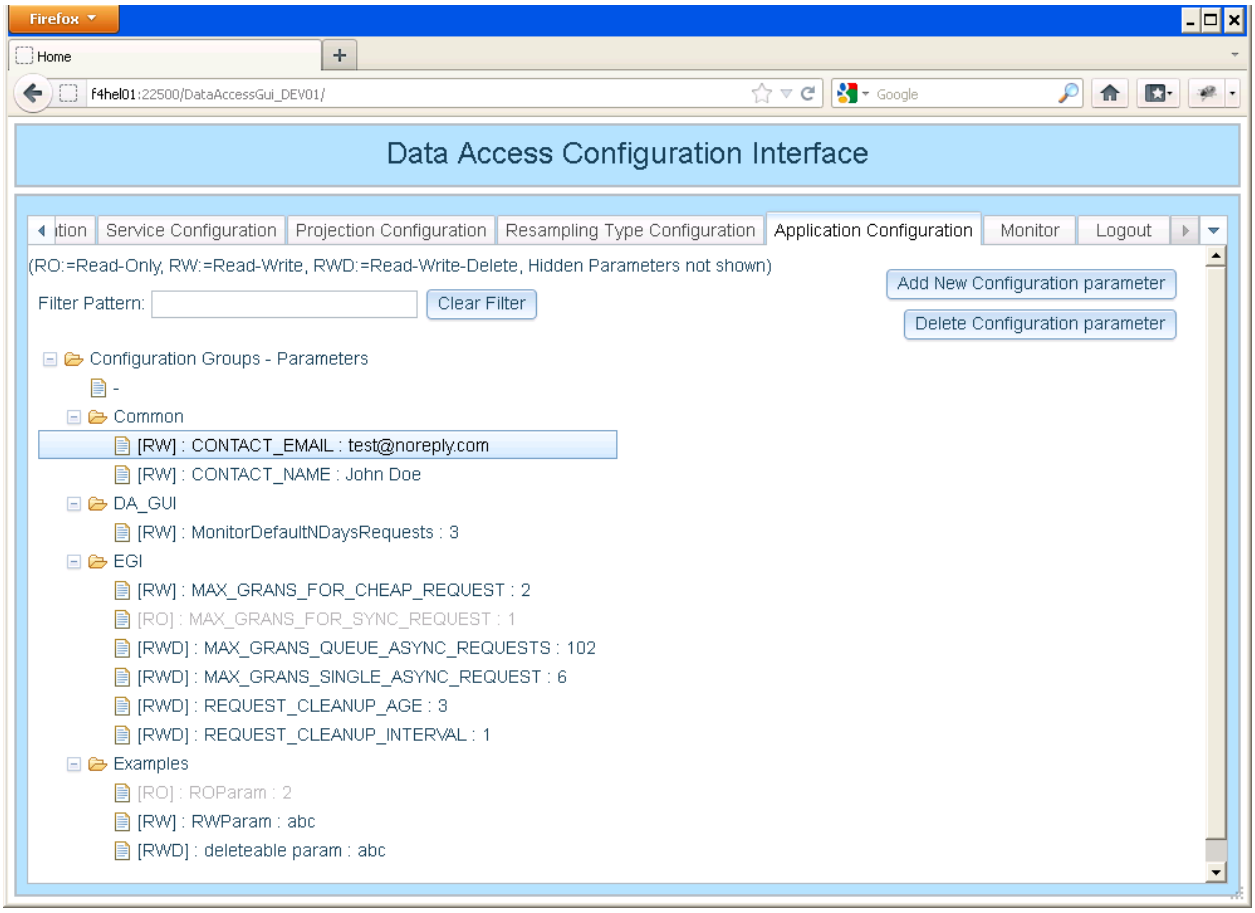


Figure 4.8.22-29. Data Access GUI Application Configuration Page

Note that Application Parameters may be pre-configured in the database. Application parameters have a database attribute that affects their behavior in the GUI. The “permissions”

attribute establishes whether the parameter is *Hidden* (does not even appear in the GUI), is *Read-Only*, is *Read-Write* or is fully accessible and *Deleteable*. Any parameter that is created in the GUI has this latter permissions setting, and only a *Deleteable* parameter can be deleted from the GUI.

Double-Clicking on an application parameter brings up the Configure Application Parameter dialog. This offers the Name, Group, Description, Data-Type and Value fields. Note that a *Read-Write* parameter refers to the read-write status of the parameter value, not to the other attributes of the parameter. For a *Read-Write* parameter, the value is editable while all other attributes are disabled for editing. Note also that the Save-and-Close button must be pressed to save any changes made on this dialog.

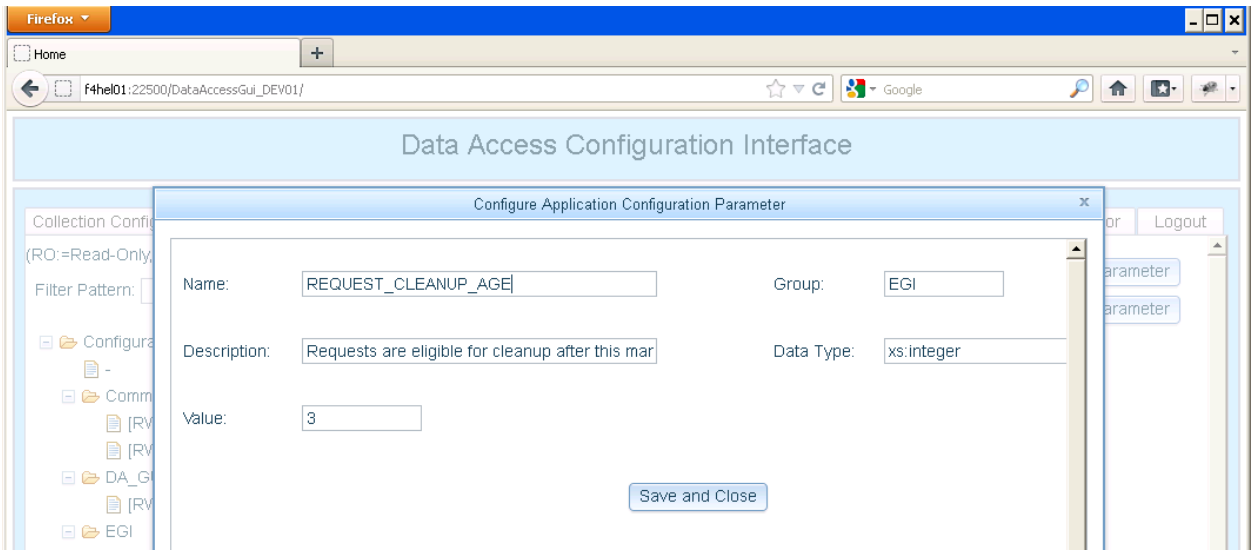


Figure 4.8.22-30. Data Access GUI Application Configuration Configure Parameter Page

4.8.22.1.6.1.8 Monitoring ESI Requests and Jobs

The monitor tab presents the user an interface to monitor ESI requests and jobs. The table on the top-left is a listing of services against which requests may be made. The table to the bottom-left is a list of requests. The fields and button in the middle affect the filling of the requests table. Shown are two dates for beginning and ending of the date-range. By default this is the current date and <n> days before, where n is a configuration parameter on the previous Application-Configuration tab (MonitorDefaultNDaysRequests).

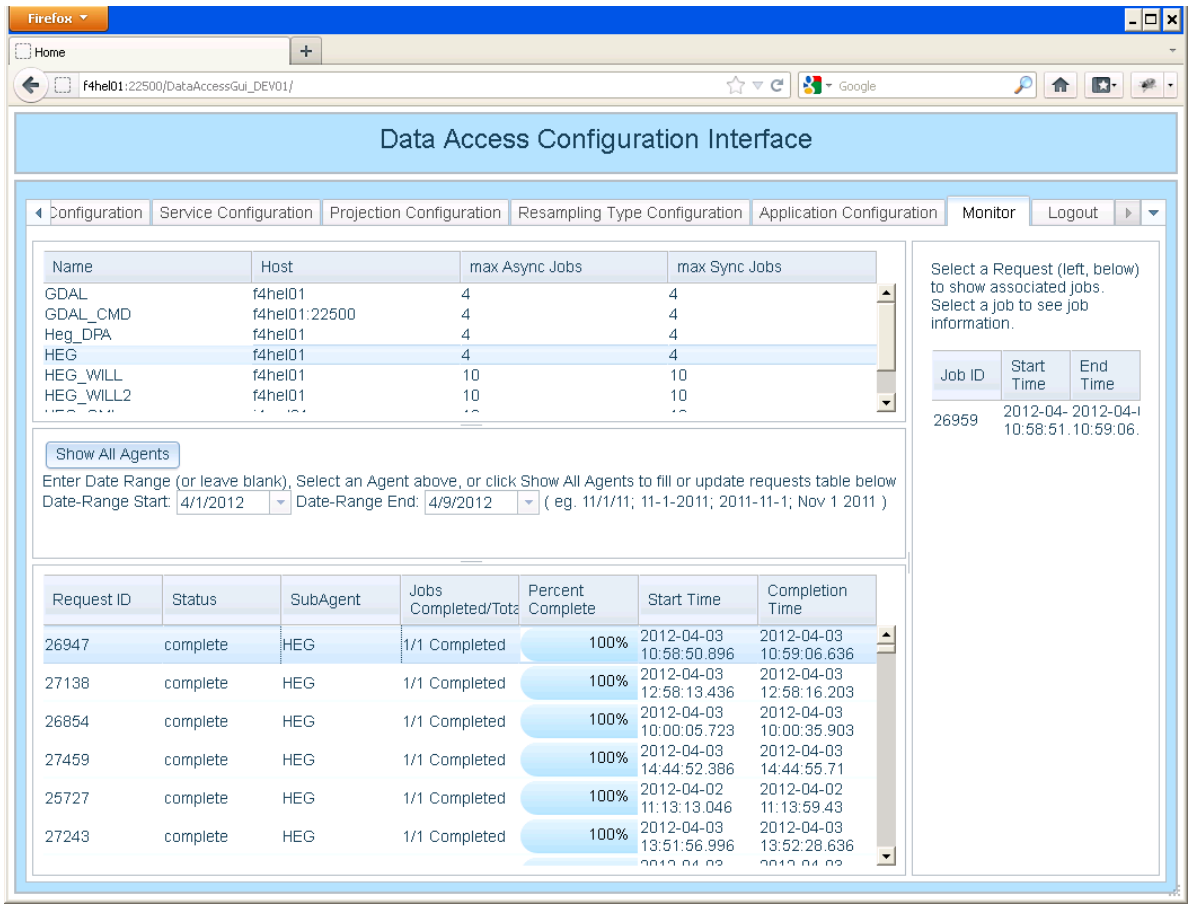


Figure 4.8.22-31. Data Access GUI Monitor Page

Clicking on the service entry in the table above, or on the Show-All-Agents button in the middle with populate the requests table below, either with just requests for that service/agent or for all agents respectively. Then, clicking on a request will show the specific jobs associated with that request in the table to the right. There is no 'auto refresh' feature, so the user will need to click on the service of interest, or in the SHow-All-Agents button in order to pick up any new requests since the page was last rendered.

Note that a given request may have one or more jobs created, one each for each granule specified in the request. Many requests have only one granule, and internal calls from ESI to an agent are for only one granule at a time – this is the definition of a job. An EGI request, however, may have multiple granules specified, and processing for these granules is implemented as separate ESI jobs. Note also that each job – a per-granule-request – may have multiple output files. The job reflects a single input granule but may have multiple output files.

Hovering over a job in the jobs table will show the success/failure status and the number of files generated. Double-Clicking on a job will bring up a dialog showing the job's URL, status, file-count, and the error-code and error-message. Note that the last two of these – the error-code and error-message will be blank if the job completed successfully.

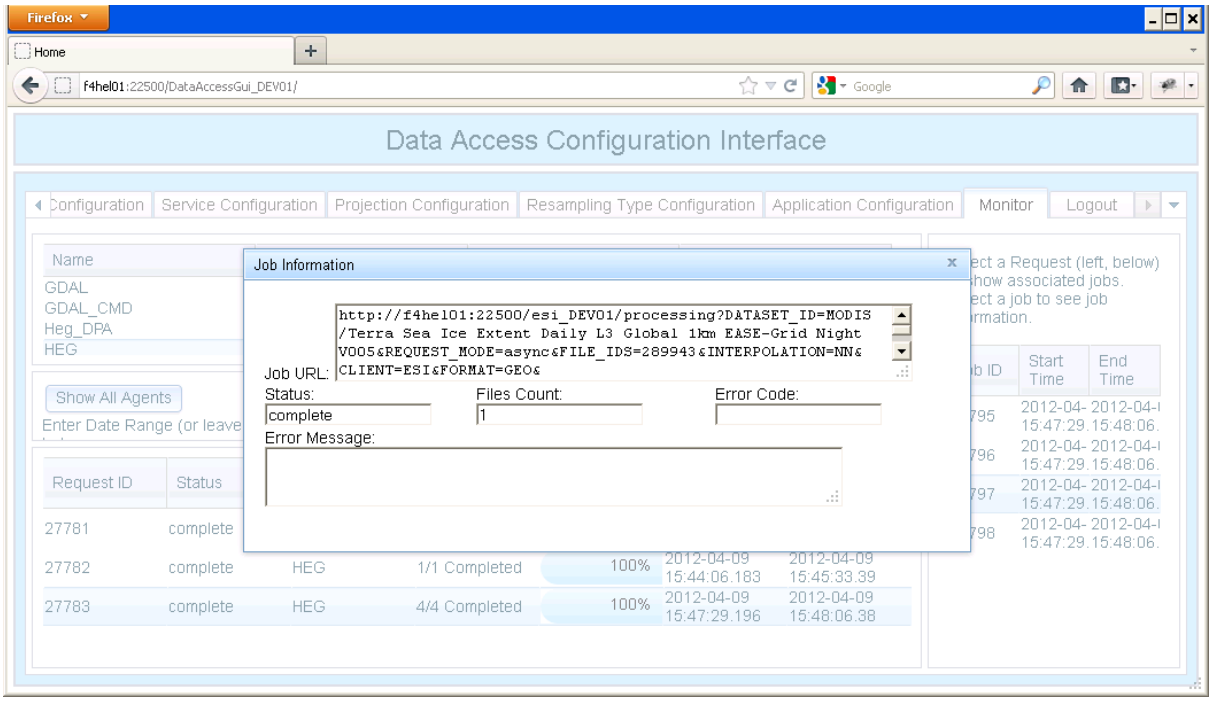


Figure 4.8.22-32. Data Access GUI Monitor Job Information Page

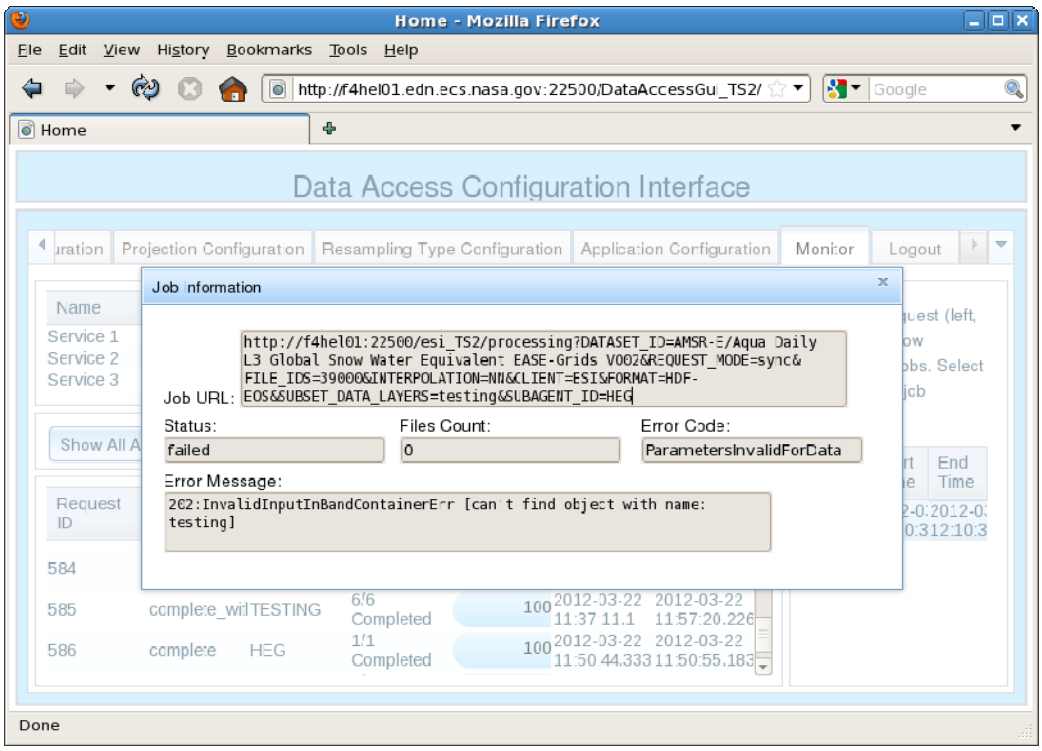


Figure 4.8.22-33. Data Access GUI Monitor Job Information Page for Job with Errors

4.8.22.1.6.2 Configuration for the Data Access GUI

The Data Access GUI gets all configuration settings as context-parameters defined in web.xml. The web.xml file is located in the /usr/ecs/<mode>/CUSTOM/WWW/DPL/DataAccessGui/WEB-INF directory. Table 4.8.22-15 describes the configuration parameters.

Table 4.8.22-17. Configuration Parameters – web.xml

Parameter Name	Value Description
mode	System Mode, e.g., TS2 or OPS
ecsHome	ECS home/root directory, typically /usr/ecs
binDir	ECS Mode bin directory, typically /CUSTOM/bin/COM
connDriver	Java class name for Sybase database connection
dbHost (SYBASE_HOST in ECS Assist)	Host machine name for the database
dbPort (SYBASE_PORT in ECS Assist)	Host port number for accessing the database
dbName	Name for the database containing Data-Access-GUI data.
dbUserName	Account name for accessing Data-Access-GUI tables in database.
progId	Program Id used in accessing the database
cryptoName	Script (in binDir) for accessing the database password
role-name (DAGUI_ACCESS_ROLE in ECS Assist)	This parameter (under security-role and security-constraint) defines the security role that a user must possess in order to access the GUI if authentication is enabled.
url-pattern (SECURED_URL_PATTERN in ECS Assist)	This parameter (under security-constraint) defines a pattern which selects what pages in the GUI to put under authentication control. The default is '/none', which effectively disables authentication on the GUI. In order to enabled authentication, this value should be set to '/'

Table 4.8.22-18. Application Configuration Values Set in the "Application Configuration" Tab of the GUI

Parameter Name	Value Description
CONTACT_EMAIL	Contact email address included in the response sent to user.
CONTACT_NAME	Contact name included in the response sent to user.
MAX_GRANS_FOR_CHEAP_REQUEST	The maximum number of granules which a request can contain and still be considered a 'cheap' request. Cheap requests get priority on the job queue.
MAX_GRANS_FOR_SYNC_REQUEST	The maximum number of granules allowed in a synchronous request. This is hard coded to 1.
MAX_GRANS_QUEUE_ASYNC_REQUESTS	The maximum number of jobs that can be queued at a time.
MAX_GRANS_SINGLE_ASYNC_REQUEST	The maximum number of granules that are allowed in a single asynchronous request.
REQUEST_CLEANUP_AGE	Number of days after which to cleanup request output.
REQUEST_CLEANUP_INTERVAL	Interval, in hours, on which to check for output to cleanup.
MonitorDefaultNDaysRequests	Default number of days worth of requests to show in the Monitor tab.

4.8.22.1.6.3 Main Screen

The Data Access GUI screens are outlined in the previous sections.

4.8.22.1.7 getfile web service

The getfile webservice is a very simple web service which provides HTTP access to files under the /datapool filesystem. This service is intended to be used by tool adapters and other entities running within the DAAC firewall but without direct access to the granule files under /datapool. This facilitates distribution of ESI back end processing.

getfile is intended only to be run on hosts that are within the DAAC firewall and not accessible to the external Internet. getfile has some security built in; only files under /datapool can be downloaded and hidden granule files cannot be downloaded, but otherwise provides wide open access to all files under /datapool and therefore is intended for internal use only. For external access to granule files, use the ESI Result Interface (ESIR) which only allows access to files associated with ESI requests and hides the actual physical location of the downloaded files.

It is worth noting that the getfile service is essentially a very simple ESI internal tool adapter. However, it does not follow the full ESI API (it only accepts the FILE_URLS parameter) and replies with a binary stream of the requested file, rather than an ESI compliant XML response.

4.8.22.1.7.1 Usage

The getfile interface is a web service which is invoked via an HTTP GET request. The URL for getfile is as follows:

http://host:port/getfile(_MODE)?FILE_URLS=<filePath>

Where <filePath> is the full path to the file to be downloaded. This path must begin with '/datapool'

4.8.22.1.7.2 Examples

http://f4hel01:22500/getfile_DEV01?FILE_URLS=/datapool/DEV01/user/FS2/MOLT/MOD13A1.005/2011.01.01/MOD13A1.A2011001.h00v09.005.2011025172858.hdf.xml

4.8.22.1.7.3 Configuration

The getfile interface does not have any configuration parameters.

4.8.22.2 External Interfaces

4.8.22.2.1 ESI Gateway Interface (EGI)

The ESI Gateway Interface (EGI) is a webservice which runs on an external host .

4.8.22.2.1.1 Using the ESI Gateway Interface

The ESI Gateway Interface (EGI) is a webservice which can be invoked via an HTTP GET or POST request. EGI will usually be called via the ECHO REVERB Client. The URL for EGI is as follows:

http://host:port/egi(_MODE)/

To make a capabilities request you can use the following URL:

http://host:port/egi(_MODE)/capabilities/<ShortName.VersionId>

Where <ShortName.VersionId> is the shortname and versionid of the collection you are interested in.

For a processing request you can use the following URL:

http://host:port/egi(_MODE)/request

The processing parameters can be passed in via HTTP GET (ie appending ?param1=value1,param2=value2,... to the URL) or via HTTP POST. The following is a table of the parameters available:

Table 4.8.22-19. Request Parameters (1 of 2)

Parameter Name	Required	Description
SUBAGENT_ID	No	Identification of desired agent to perform internal processing. If this agent is unavaiable or unable to perform the requested processing on the requested file, the request will result in an error. If no value is specified, then ESI will be responsible for choosing a tool to forward the request to.
REQUEST_MODE	No	The type of request that is being made: synchronous or asynchronous. A synchronous request will result in a response from the agent only when the request has been processed to completion. An asynchronous request will result in a response once the request has been registered in an internal database and queued for processing. Some agents will only allow one type or the other, as specified in the agent configuration, and the specification of an unsupported request mode in the request parameters will result in an error. This parameter is necessary since ESI allows an agent to provide both synchronous and asynchronous services, unlike SSW.
OUTPUT_DIR	No	Local path of directory where output file should be placed. This parameter is only used if the agent (tool adapter) and the external web API have access to common storage (e.g. a SAN file system).
PROJECTION	No	Output projection for the subsetted file
PROJECTION_PARAMETERS	No	Comma separated name:value pairs of parameters to be used by the output projection. This parameter is only valid if a PROJECTION is also specified.
RESAMPLE	No	The size of pixels in the output file. In the format of [DIMENSION]:[size]
INTERPOLATION	No	The interpolation method to use for rescaling and/or reprojecting of the output data
CLIENT	No	The API used for this request. Valid value are "SSW" and "ESI". This will determine if the server should use SSW or ESI syntax for validating the request. The default value will depend on the particular server, but will need to be "SSW" for any server which intends to be backwards compatible with SSW (since and SSW client will not provide this parameter).
DATASET_ID	No	ECHO DataSetID value (collection title)
START	No	Subset start date/time in RFC-3339 format
END	No	Subset end date/time in RFC-3339 format
BBOX	No	Subset bounding box: sequence of values in decimal degrees, separated by commas, in the order west,south,east,north
SUBSET_DATA_LAYERS	No	Comma-separated variable names
FORMAT	No	Format of subsetted file

Table 4.8.22-19. Request Parameters (2 of 2)

Parameter Name	Required	Description
NATIVE_PROJECTION	No	Native projection of the subsetted file
OUTPUT_GRID	No	Output grid of the subsetted file
EMAIL	No	E-mail address for asynchronous subsetting response
FILE_IDS	Yes	The granule identifier(s) obtained from the atom:id tags of the granule OpenSearch response. A comma-separated list if there are multiple granules and the granule_multiplicity for the agent is "batch" or "both".
FILE_URLS	No	The data URL identifier(s) obtained form the link tags with a rel attribute equal tohttp://esipfed.org/ns/fedsearch/1.0/data#in the granule OpenSearch response. A comma-separated list if there are multiple granules and the granule_multiplicity for the agent is "batch" or "both".

4.8.22.2.1.2 ESI Gateway Interface Examples

Submit a request for HEG processing of two granules as an HTTP GET request:

http://f4eil01:22500/egi_DEV01/request?FILE_IDS=249600%2C250235&CLIENT=ESI&REQUEST_MODE=sync&DATASET_ID=MODIS%2FTerra%20Snow%20Cover%20Monthly%20L3%20Global%200.05Deg%20CMG%20V005&INTERPOLATION=NN&FORMAT=HDF-EOS&SUBAGENT_ID=HEG&

Submit the same request for HEG processing of two granules as in the above URL, but as an HTTP POST request:

wget -O - --server-response --post-data 'FILE_IDS=249600%2C250235&CLIENT=ESI&REQUEST_MODE=sync&DATASET_ID=MODIS%2FTerra Snow Cover Monthly L3 Global 0.05Deg CMG V005&INTERPOLATION=NN&FORMAT=HDF-EOS&SUBAGENT_ID=HEG&' http://f4eil01:22500/egi_DEV01/request'

Notice that in both of the preceding examples, special characters are escaped using HTTP URL escaping rules. In many cases, this escaping is not strictly required, depending on how your HTTP client handles special characters, but it is recommended to always escape them to avoid confusion or errors. http://www.w3schools.com/tags/ref_urlencode.asp contains a list of rules for escaping special characters.

Get the current status (and download URLs if any) for already submitted request 24357:

http://f4eil01:22500/egi_DEV01/request/24357

Get the ESI DataSet XML specifying what processing is available for collection AE_SI25.002:

http://f4eil01:22500/egi_DEV01/capabilities/AE_SI25.002.xml

Get the ECHO for, which when uploaded in the ECHO system will provide ESI order options for collection AE_SI25.002:

http://f4eil01:22500/egi_DEV01/forms/echo/AE_SI25.002.xml

4.8.22.2.1.3 ESI Gateway Interface GUI Configuration

The ESI Gateway Interface uses properties from the Database edited via the GUI. (See GUI above for more info) On the Application Properties Tab of the GUI, you will find a section for EGI. Table 4.8.22-17 describes these properties.

Table 4.8.22-20. Configuration Parameters

Parameter Name	Value Description
MAX_GRANS_FOR_CHEAP_REQUEST	The maximum number of granules in a request for it to be considered a “cheap” request(ie not use too many resources).
MAX_GRANS_FOR_SYNC_REQUEST	The maximum number of granules allowed in a synchronous request.
MAX_GRANS_QUEUE_ASYNC_REQUESTS	The maximum number of granules allowed in the async queue.
MAX_GRANS_SINGLE_ASYNC_REQUEST	The maximum number of granules allowed in a single asynchronous request
REQUEST_CLEANUP_AGE	The number of days in which a request is considered eligible for cleanup.
REQUEST_CLEANUP_INTERVAL	The number of hours between request cleanup calls.

4.8.22.2.1.4 ESI Gateway Interface Configuration File

The ESI Gateway Interface uses a properties file, EcDlDaEgi.properties, 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 4.8.22-18 describes the configuration parameters.

Table 4.8.22-21. Configuration Parameters

Parameter Name	Value Description
application.name	Name of this application.
program.id	ID 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.
esi.processing.url	The URL for ESI processing.
esi.capabilities.url	The URL for ESI capabilities.
esi.url	The URL for ESI.
esir.url	The URL for ESIR.
dataaccess.email.from	The email address which emails can be replied to.
mail.smtp.host	The email SMTP host.
ftp.host	The FTP host.

4.8.22.2.1.4 ESI Gateway Interface Main Screen

The ESI Gateway Interface does not have a main screen. It is a webservice. Please see EGI Usage above.

4.8.22.2.2 ESI Result Interface (ESIR)

The ESI Result (ESIR) Interface webservice provides access to the results of ESI processing via HTTP download. This interface is opened to the external network, but limits access to only those files which are associated with a valid ESI request.

4.8.22.2.2.1 Using the ESI Result Interface (ESIR)

The ESI Result Interface (ESIR) is a web service that can be called via the HTTP GET method. The URLs for ESIR are as follows:

http://host:port/esir(_MODE)/{requestId}
http://host:port/esir(_MODE)/{requestId}.html

Where {requestId} is the identifier of a completed request. Both URLs will return an HTML page containing a list of URLS to download all files associated with the request. This includes all science/image files, any metadata or ancillary files, and a zip file that contains all of these in a single file.

http://host:port/esir(_MODE)/{requestId}.zip

Where {requestId} is the identifier of a completed request. This URL will return a zip file, generated on demand, which contains all of the files associated with the request. The files that will be included in the zip file are those which are listed in the HTML file listing page (mentioned above). The zip file will have a separate sub directory for each input granule that has output files.

http://host:port/esir(_MODE)/{requestId}/{granuleId}/{fileName}

Where {requestId} is the identifier of a completed request, {granuleId} is the ECS GranuleId of the input granule that resulted in the generation of the requested file, and {fileName} is the file basename (i.e. the filename with no directory path) of the requested file. The {granuleId} is necessary to avoid collisions when multiple input granules to a multiple granule request generate output files with the same name. In general, a user would not construct such a URL manually, but would be given the URL either in the HTML file list mentioned above, or in the result XML or email delivered to the user after request completion.

4.8.22.2.2.2 ESIR Examples

Display a list of links to the files generated by request 24357:

http://f4eil01:22500/esir_DEV01/24357.html

Automatically generate and download a zip file containing all output files generated by request 24357:

http://f4eil01:22500/esir_DEV01/24357.zip

Download a GeoTiff image file generated by request 24357:

http://f4eil01:22500/esir_DEV01/24357/289982/MOD13A1_A2011001_h00v09_005_2011025172858_500m_16_days_NDVI_8639cbb4.tif

4.8.22.2.2.3 Configuration

The ESI Result Interface uses a properties file, EcDlDaEsir.properties, 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 1.1.1.2.2-1 describes the configuration parameters.

Table 4.8.22-22. Configuration Parameters

Parameter Name	Value Description
APP.ID	ID of this application.
LOG.FILE	Log file. If absent the log messages will be sent to the console (i.e. tomcat log file). The token "<timestamp>", if included in log file name, will be replaced with yyyyMMdd timestamp of the log file creation.
LOG.LEVEL	Logging level. This must be one of: SEVERE (highest), WARNING, INFO, CONFIG, FINE, FINER, FINEST, OFF, ALL
SYBASE.JDBC.DRIVER.CLASS	Sybase JDBC driver
SYBASE.JDBC.URL	Sybase Server URL
SYBASE.USER	The Sybase user which is to be used by ESIR (EcDlDaDataAccess)
DCRP.BIN.PATH	Location of the password decryptor utility
MODE	The ECS MODE
ZIP.STAGING.AREA	Directory where the zip files will be created (double slashes are needed in place of directory separator slashes).

4.8.22.2.2.4 Main Screen

The ESI Result Interface does not have a main screen. It is a web service. Please see ESIR Usage above.

4.8.22.3 Integrating a Custom Tool Adapter

The Data Access system is designed to facilitate the integration of any number of data processing services. The default installation contains the HegService, GdalService, and GDAL Command Line Adapter (as an example only) as built in services. However, it is fairly straightforward to add additional services with minimal or no changes to the Data Access software baseline.

4.8.22.3.1 Implement an ESI Tool Adapter

An ESI tool adapter is a software entity that accepts inputs conformant to the ESI API, and translates into the input syntax of a processing tool. Such an adapter will allow a processing tool to be integrated into the Data Access system that will allow it to be available for processing requests from ECHO Reverb and other clients.

The ESI API is derived from, and therefore quite similar to the Simple Subset Wizard (SSW) API, and therefore, existing SSW implementations can be used as ESI Tool Adapters without much extra effort. The ESI API is documented at http://newsroom.gsfc.nasa.gov/esi/8.1/schemas/doc/req/eesi/index.html. Note that while this documentation is written as an XML schema, the tool adapter should be implemented to accept

request parameters as query parameters in an HTTP GET request. The elements in the XML schema referenced above correspond to the key names of the URL query parameters.

A tool adapter can be written one of two ways.

1) Directly as a webservice accepting HTTP GET requests which contain query parameters.

2) As a command line script accepting requests as a command line invocation in the form <script path> --<param1name> <param1value> --<param2name> <param2value>

If option 2 is chosen, the REST Quick Server (RQS) will be used to allow a REST style web service interface to the command line script. See the RQS section of this document for more on how to configure the RQS for a given command line adapter.

If option 1 is chosen, there are many frameworks available for implementing REST style webservices with minimal effort. We recommend using the Sun Jersey framework, which was used extensively in the development of Data Access components. Another option is the Spring framework, but it is more complex than Jersey and therefore has a steeper learning curve. Below is the Java code to create a simple web resource in Jersey which accepts a GET request and parses the query parameters. See the Jersey documentation (at http://jersey.java.net) for more information on how to set up this resource within a web application server (such as Tomcat).

```
//defines the path that this class responds to relative to the application's base URI
@Path("/example/{token}.{ext}")
public class SampleRestResource {
        //says that this class will respond to a GET request with an HTML response
        @GET
        @Produces("text/html")
        public Response getResponse
           (@PathParam("token") String token,
            @PathParam("ext")   String ext,
            @Context            UriInfo ui)
        {
                //params = map of all query parameters
                //token and ext are parts of the URL as shown in the @Path annotation
                //ui has some context info.
                MultivaluedMap<String, String> params = ui.getQueryParameters();
                String baseUri = ui.getBaseUri().toString();
                String content = "<html><body>";
                content += "<h1>Congratulations!!</h1>";
                content += "The Webservice is running correctly<br/>";
                content += "Base URI for this service is: " + baseUri +"<br/>";
                content += "Token passed in at end of URL = " + token +"<br/>";
                content += "Values of all query parameters: " + "<br/>";
                content += params.toString().replace("],","]<br />");
                content += "</body></html>";
                String mt = "text/html"; // set the mime type of the response
                return Response.ok(content, mt).build();
        }
}
```

The adapter can use, if necessary, the Simple Inventory Interface to obtain inventory information about the granules being processed and getfile webservice to obtain granule files, if it is running on a machine without access to the datapool filesystem.

A tool adapter can expect to always receive:

- GranuleId of an input granule in the FILE_IDS field and the datapool path of the granule file in the FILE_URLS field. The adapter can choose which one of these to use to specify the input data.
- A single granule per invocation.
- Request parameters as defined in http://newsroom.gsfc.nasa.gov/esi/8.1/schemas/ESIRequestInternal.xsd / http://newsroom.gsfc.nasa.gov/esi/8.1/schemas/doc/req/iesi/index.html and as configured for the associated service in the Data Access GUI. The adapter may not be able to process all the request parameters specified in the above linked documentation, but should at least be able to accept all of them without failure (in case an extraneous parameter makes it into a request).

A tool adapter shall:

- Invoke processing on the specified granule, and wait to the processing to terminate.
- Respond with an XML response conformant to the schema documented at http://newsroom.gsfc.nasa.gov/esi/8.1/schemas/doc/rsp/iesi/index.html and http://newsroom.gsfc.nasa.gov/esi/8.1/schemas/ESIAgentResponseInternal.xsd.
- Include in the response downloadUrls which are either file system paths to a location accessible to the EGI service, or HTTP URLs allowing EGI to obtain the output files. The getfile webservice could be used by a tool adapter to expose output files as HTTP URLs.
- Clean up its temporary files (but not the output files) either immediately, or after a reasonable amount of time. The EGI will be responsible for cleaning up the output files on a configurable schedule.

The EGI service will implement a time out for each service, which will terminate the HTTP connection from the client's end. This may not terminate the process on the server however. So, it is recommended that the tool adapter implement its own timeout, which is greater than or equal to that configured in EGI for the same service.

Below is an example response from a tool adapter:

```
<iesi:agentResponse xmlns:iesi="http://eosdis.nasa.gov/esi/rsp/i"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:esi="http://eosdis.nasa.gov/esi/rsp"
xmlns:ssw="http://newsroom.gsfc.nasa.gov/esi/rsp/ssw"
xmlns:eesi="http://eosdis.nasa.gov/esi/rsp/e"
xsi:schemaLocation="http://eosdis.nasa.gov/esi/rsp/i
http://newsroom.gsfc.nasa.gov/esi/8.1/schemas/ESIAgentResponseInternal.xsd">
    <downloadUrls>

<downloadUrl>/datapool/TS2/user/FS1/Gdal_DefaultOutput/d467914a_5/MOD10CM.A2003182.005
.2007348190637_MOD_CMG_Snow_5km_Snow_Cover_Monthly_CMG.jpg</downloadUrl>

<downloadUrl>/datapool/TS2/user/FS1/Gdal_DefaultOutput/d467914a_5/MOD10CM.A2003182.005
.2007348190637_MOD_CMG_Snow_5km_Snow_Spatial_QA.jpg</downloadUrl>
```

```
    </downloadUrls>
    <contactInformation/>
    <processInfo>
        <message>Success!</message>
        <processDuration>PT4.596S</processDuration>
        <internalCommand>EsiRequest: Format= JPEG | Subset_Data_Layers= [] |
Include_MetaData= false | Projection= null | Projection_Parameters= {} | BBox= null |
Interpolation= null | Resample= null | File_Id= 384557</internalCommand>
        <processIdentifier>d467914a_5</processIdentifier>

<inputFile>/datapool/TS2/user/FS1/MOST/MOD10CM.005/2003.07.01/MOD10CM.A2003182.005.200
7348190637.hdf</inputFile>
        <outputDisposition>ExternallyAccessible</outputDisposition>
    </processInfo>
    <requestStatus/>
</iesi:agentResponse>
```


4.8.22.3.2 Define a Service in Data Access GUI

Once a tool adapter has been implemented, it can be integrated into the Data Access system via the Data Access GUI, following the instructions in the Data Access GUI section of this document. Once configured in the GUI, it is possible to specify all of the valid processing options for the processing tool in question. Be sure to limit the options to those which the tool is capable of processing. Once the configuration has been updated, be sure to generate and upload new ECHO forms to the ECHO system so that the new service is available via ECHO. Changes will show up in the Dataset Config XML document and ESI drilldown webpages immediately.

4.8.22.4 Required Operating Environment

The EOSDIS Service Interface (Data Access) runs on a Linux platform.

4.8.22.5 Interfaces and Data Types

Table 4.8.22-20 lists the supporting products that this tool depends upon in order to function properly.

Table 4.8.22-23. Interface Protocols

Product Dependency	Protocols Used	Comments
Data Pool database	SQL	Via SQL server machines
Inventory database	SQL	Via SQL server machines
Java JRE version 1.6.0_02	Linux system call	
Java jConnect	Java Library	
Apache Tomcat	Web Server	Web Server used to deploy the different Web Services

4.8.22.6 Outputs

The EOSDIS Service Interface (Data Access) generates processed ECS Data. The processing services performed on the data depends on user input and configured available services. The data can be downloaded via HTTP. Downloads are usually handled via the ESIR (See ESIR above.)

4.8.22.7 Event and Error Messages

Error messages will be displayed in the log files and returned via the HTTP Response object for HTTP calls for the different web services or Standard Out for command line tool adapters.

4.8.22.8 Logs

Each Data Access component has its own log file which is located in the /usr/ecs/<mode>/CUSTOM/logs directory. The log file will contain the initial call to the component, including all arguments, as well as all database stored procedure calls, any errors that occur, and other pertinent information. Fatal errors are printed to the log. Errors that occur before the log is opened will be printed to the tomcat log.

4.8.22.9 Recovery

The Data Access Components (EGI) can recover from interruptions caused by situations such as the system faults, leaving some requests unfulfilled. To recover, EGI will look up all uncompleted requests asynchronous jobs and requeue them. All synchronous jobs that were not completed will be failed.

4.8.22.10 Sybase Error Handling

If a Sybase error occurs, the actual Sybase error string will most likely be displayed on the screen or 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.

This page intentionally left blank.

4.8.23 DataPool Checksum Verification Server

The DataPool Checksum Verification ServerServer (CVS) provides a mechanism by which the ECS Operations Staff can perform checksum verification for files in the Data Pool. CVS retrieves a list of granules to checksum from the DataPool database DlCvRequest table, which are entered via theEcDlInsertChecksumRequest.pl script (see instructions below). The CVS is able to efficiently checksum files in the DataPool, verify the checksum's that the DPL DB has for files in the DataPool, as well as change the checksum type/origin of files in the DataPool, all triggered via the records in the DlCvRequest table.

- CVS searches for records in DlCvRequest that have a Status of "N", and for each DplId (granuleId equivalent) a checksum verification is done, using whichever checksum type (retrieved from the DlFile table) that was previously used for that granuleId. If a granule's ChecksumType and ChecksumOrigin fields are populated in the DlCvRequest table, CVS will checksum the file(s) associated with that granuleId according to the specified ChecksumType and will change the file(s) checksum origin and checksum type in the DlFile table to the checksum type and origin specified in the DlcvRequest table.
- Upon successful checksum verification, CVS will update the time when checksum was verified for each file in the DataPool database. CVS will also mark the requests in DlCvRequest table as complete by setting the Status field to "C".
- Upon detection of checksum verification failure, the utility will log detailed information about the failure which will include granule ID, ESDT, insert time, complete file path and file name, along with the checksum information -- including checksum type, checksum values (computed value vs. the corresponding value stored in database), the last time the file was checksummed, checksum origin (who performed the last checksum).
- The primary use case of CVS is to run as a service in the background, performing checksum verifications as it finds new checksum requests in the DlCvRequest table, which are currently entered via a user running the EcDlInsertChecksumRequest.pl script from the command line.
- Following the standard of most EMD servers, the CVS will not allow multiple instances of itself to run in a particular mode.

4.8.23.1 Using the DataPool Checksum Verification Server

In order to trigger the checksuming process by the CVS, the user must first populate the DlCvRequest table with checksum requests. This is done by running the EcDlInsertChecksumRquest.pl script. To run this script, do the following:

1 Log in at the machine where the Data Pool Checksum Verification Utility is installed (e.g., x4dpl01).

2 To change to the directory for starting the Data Pool Checksum Verification Utility, type **cd /usr/ecs/<*MODE*>/CUSTOM/utilities** and then press the **Return/Enter** key.

- The working directory is changed to **/usr/ecs/<*MODE*>/CUSTOM/utilities**.

3 At the prompt, type the command to start the Data Pool Checksum Verification Utility, in the form of the following.

EcDlInsertChecksumRequest.pl –mode <MODE> [-file|f filename] [-granule|g id] [-esdt esdtname] [-profile name] [-filesystem|fs dplfilesystem] [-priority priority] [-batchlabel batch] [-help|h]

Example of Data Pool Checksum Verification Service Client:

- **EcDlInsertChecksumRequest.pl -mode OPS -g 12345**
  Verify checksum for Data Pool granule 12345 in OPS mode

- **EcDlInsertChecksumRequest.pl -mode OPS -esdt AST_L1A.004**
  Verify checksums for all granules from collection AST_L1A.004 in OPS mode

- **EcDlInsertChecksumRequest.pl -mode OPS –file /home/cmshared/test.dat**
  Verify checksums for all granules specified in file /home/cmshared/test.dat

Once the above steps have been completed, the user may Run the DataPool Checksum Verification Server. The DataPool Checksum Verification Server should be started by the user cmshared (or similar). The DataPool Checksum Verification Server is started by entering the following command:

EcDlCVSStart <mode>

There is one command line parameter that may be used. Table 4.8.23-1 provides a description of the parameter.

Table 4.8.23-1. Command Line Parameter

Parameter Name	Required	Description
mode	yes	Mode in which the Checksum Verification Server should be ran.

When running EcDlCVSStart without any parameter on the command line, it will print out its usage, as follows:

Usage: EcDlCVSStart MODE [param1 value1] [param2 value2] ...

(NOTE: any command line parameters following the mode are ignored by the server)

4.8.23.1.1 DataPool Checksum Verification Server Command Line Examples

1. To checksum all science files associated with the granuleId's found in DlCvRequest table with Status='C' run:

```
EcDlCVSStart OPS
```

The DataPool Checksum Verification Server will perform checksum verification for all granule files associated with the granuleId's found in the DlCvRequest table with Status='C'.

4.8.23.2 DataPool Checksum Verification Server Configuration File

The DataPool Checksum Verification Server uses a configuration file: EcDlDpcv.properties, 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 4.8.23-2 describes the configuration parameters.

Table 4.8.23-2. Configuration Parameters

Parameter Name	Description
AppLogSize	Maximum size that the application log can be before starting a new application log
AppLogLevel	Level of detail to be logged in the application log
DebugLevel	Level of detail to be logged in the debug log
DPL_SERVER	Sybase server name for the DataPool database connection
DPL_DATABASE	DataPool database name (ex: DataPool_TS1..)
DPL_USER	The user name for the DataPool database Sybase connection.
DPL_PASSWDSEED	Seed used to dynamically generate a DB login password for the server
DPL_MinPoolSize	The minimal size of the DPL database connection pool.
DPL_MaxPoolSize	The maximum size of the DPL database connection pool.
DPLChecksumServiceHostPort	Host Port that the CVS reads from
DPLChecksumServiceHosts	Hosts that the server may be ran on
DlCChecksumRequestRetryLimit	How many times to retry on a "retryable" error (such as DB connection failure etc…) – default is 5
DlCChecksumRequestRetryInterval	How long to wait before retrying a failed operation

4.8.23.3 DataPool Checksum Verification Server Main Screen

The DataPool Checksum Verification Server does not have a main screen. It has a command line interface only.

4.8.23.4 Required Operating Environment

The DataPool Checksum Verification Server will run on a Linux platform.

4.8.23.5 Databases

Table 4.8.23-3 lists the supporting products that this tool depends upon in order to function properly.

Table 4.8.23-3. Product Dependencies

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

4.8.23.6 Special Constraints

The DataPool Checksum Verification Server runs only if the Datapool database and Inventory database server is running and if the databases are available. It also assumes the stored procedures are present, EcDlActionDriver is the user it logs into the DB as and is a registered user in both databases and have proper permission to execute the stored procedures.

4.8.23.7 Outputs

CVS generates both an Application and Debug log file (See Section 4.8.23.10 for details). The main "output" of this server is the changes that it makes to the DlFile table with regard to the granules' files being checksum processed.

4.8.23.8 Event and Error Messages

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

4.8.23.9 Reports

See outputs above.

4.8.23.10 Logs

CVS produces a log file called EcDlChecksumServerDebug.log and a EcDlChecksumServer.ALOG. The log file will reside in the /usr/ecs/<mode>/CUSTOM/logs directory. In addition to logging the completion of a checksum verification request, when the CVS has finished processing a request, the corresponding entry in the DlCvRequest table will be marked completed by setting the Status field to 'C'.

4.8.23.11 Recovery

If the DataPool Checksum Verification Server goes down during processing, when the operator brings the server backup it will read the DlCvRequest table and see that there is a request who's status is not complete 'C' and continue to process that request.

4.8.23.12 Sybase Error Handling

If a Sybase error occurs, the actual Sybase error string will most likely be logged 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 not be able to process any granules.

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, DlCChecksumRequestRetryLimit was set to 3 and DlCChecksumRequestRetryInterval 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.

This page intentionally left blank.

4.8.24 OM Cleanup Old Data Utility

The OM Cleanup Old Data Utility (EcOmCleanupOldData.pl) provides a basic function for ECS Operations Staff: a mechanism to archive orders in various states of completion. The processing of non-expired bundling orders is not supported.

When processing the orders, the utility will remove the database entries associated with the orders that have completed before a date specified in the OmConfig portion of the OM GUI. The relevant command line option for processing orders in a non-terminal state is: (FORCE), which is "0" by default, but can be switched to a "1" for a different mode.

The latest version of the script has been written to meet several performance standards: it is able to perform SQL queries, even in the event of deadlock; it works with PostgreSQL 9.2.

4.8.24.1 OM Old Data Cleanup Utility Configuration

The utility is located on the x5oml01 machine in the directory /usr/ecs/<mode>/CUSTOM/dbms/postgres/scripts. It uses a PostgreSQL function defined in the file "/usr/ecs/<MODE>/CUSTOM/dbms/postgres/OMS/OmCleanupOldData.sp." This PostgreSQL function depends upon configuration variables in the table "OmConfig" of the OMS schema in the ECS database. In particular, the entries, "ARCHIVE_REQUEST_AGE" and "ARCHIVE_INCOMPLETE_REQUEST_AGE" are used to determine the minimum age of requests that can be archived. The configuration parameters can be changed using the OM GUI (OM Configuration -> Cleanup Parms). The following table describes the configuration parameters applicable to Cleanup Granules.

Table 4.8.24-1. Configuration Parameters

Parameter Name	Value Description
ARCHIVE_REQUEST_AGE	Number of days until requests in a terminal state become eligble for archiving.
ARCHIVE_INCOMPLETE_REQUEST_AGE	Number of days until requests in a non-terminal state become eligble for archiving.

4.8.24.2 Using the OM Old Data Cleanup Utility

Invocation of EcOmCleanupOldData.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. The second parameter determines which of the configuration parameters will be used: a "0" (the default) will cause the utility to run normally, only archiving completed requests older than the number of days specified by ARCHIVE_REQUEST_AGE; a "1" will cause the utility to run in force mode, where it will archive all requests whose database entry, "finishDateTime," is older than the number of days specified by ARCHIVE_INCOMPLETE_REQUEST_AGE.

The usage statement for the OM Old Data Cleanup Utility is:

```
EcOmCleanupOldData.pl  <mode>
                       [<force>]
```

Table 4.8.24-2. Command Line Parameters

Parameter Name	Required	Description
mode	No	The operational mode for which the orders will be archived
force	No	Whether to archive only orders in a terminal state or not (0 or 1)

4.8.24.3 Examples:

Cleanup completed requests in the TS1 mode older than ARCHIVE_REQUEST_AGE days

EcOmCleanupOldData.pl TS1 0

Cleanup all requests in the TS1 mode older than ARCHIVE_INCOMPLETE_REQUEST_AGE

EcOmCleanupOldData.pl TS1 1

4.8.24.4 Outputs

The EcOmCleanupOldData.pl produces a log file—but no regular output files. The number outputted is equal to the number of rows that were archived.

Standard Output

```
f5oml01v{cmshared}131: perl EcOmCleanupOldData.pl OPS 1
MODE OPS  SUBSYS OMS  DSQUERY f4dbl03 5432
select OmCleanupOldData(1)
retryCounter    0
SQL-> 21
```

Log File

```
2013/08/20 13:56:43
 :26187 :  select OmCleanupOldData(1)
2013/08/20 13:56:43
 :26187 :  retryCounter    0
2013/08/20 13:56:43
 :26187 :  SQL-> 0
2013/08/20 14:08:06
 :4713 :  MODE OPS  SUBSYS OMS  DSQUERY  5432

2013/08/20 14:08:06
 :4713 :  select OmCleanupOldData(1)
2013/08/20 14:08:06
 :4713 :  retryCounter    0
2013/08/20 14:08:07
 :4713 :  SQL-> 21
```

4.8.24.5 Recovery

The EcOmCleanupOldData.pl script is designed to avoid deadlock, but is susceptible to other runtime failures, including error-ridden entries in the database (ex. misappropriated expiration dates). If a failure occurs it will retry previous order archive requests the next time it is run.

4.8.24.6 PostgreSQL Error Handling

All EcOmCleanupOldData.pl components will attempt to deal with PostgreSQL errors gracefully; if a PostgreSQL error occurs, the actual PostgreSQL error string will display on the screen and in the log.

4.8.24.7 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 EcOmCleanupOldData.log, and is stored in the /usr/ecs/<mode>/CUSTOM/logs directory.

This page intentionally left blank.

4.8.25 OM DB Clean Archive Utility

The OM DB Clean Archive Utility (EcOmDBCleanArchive.pl) provides a basic function for ECS Operations Staff: a mechanism to clean records from the OMS archive. The affected tables include: OmGranule_R, OmOrder_R, OmRequest_R, OmAddress_R, OmRequestGranPI_R, OmRequestNotes_R, and OmBundlingOrder_R.

When processing the orders, the utility will remove the archived database entries that were last updated before the given RETAINDATE, the second of two command-line arguments.

The latest version of the script has been written to use PostgreSQL 9.2, instead of Sybase.

4.8.25.1 OM DB Clean Archive Utility Configuration File

The utility uses the stored procedure, "OmCleanArchive.sp," (located in the directory "/usr/ecs/<MODE>/CUSTOM/dbms/postgres/OMS").

4.8.25.2 Using the OM DB Clean Archive Utility

Invocation of EcOmDbCleanArchive.pl should be performed by the cmshared user. The first parameter is the operational mode that contains the archives that are to be cleaned up; it is required. The second parameter, also required, determines the minimum age of archived requests that are to be deleted. The format of the date can either reflect the MM/DD/YYYY standard or the YYYY/MM/DD format used by PostgreSQL 9.2.

The usage statement for the OM DB Clean Archive Utility is:

EcOmDbCleanArchive.pl <mode> <retain_date>

Table 4.8.25-2. Command Line Parameters

Parameter Name	Required	Description
mode	Yes	The operational mode in which the orders should be archived.
retain_date	Yes	Any orders in the table "OmOrder_R" with a "timeOfLastUpdate" prior to the date given in this parameter will be removed.

4.8.25.3 Examples:

Clean archived orders last updated before August 3, 2005

EcOmDbCleanArchive.pl TS1 08/03/2005

4.8.25.4 Outputs

The EcOmDbCleanArchive.pl produces a log file—but no regular output files.

Standard Output/Log Files (same output)

```
f5oml01v{cmshared}116: perl ../dbms/postgres/OMS/scripts/EcOmDbCleanArchive.pl OPS 08/03/2005
select OmCleanArchive('08/03/2005')
NOTICE:  Total number of Orders removed from Archive tables : 928
Mon Aug 19 16:03:59 2013 : Execution of Script has Completed.
```

4.8.25.5 Recovery

TheEcOmDbCleanArchive.pl script does not have any recovery features, as any failures can be attributed to database/system configuration errors, rather than the script itself. The script can be run repeatedly without harm, in the event that a retry is needed.

4.8.25.6 PostgreSQL Error Handling

AllEcOmDbCleanArchive.pl components will attempt to deal with PostgreSQL errors gracefully; if a PostgreSQL error occurs, the actual PostgreSQL error string will display on the screen and in the log.

4.8.25.7 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 namedEcOmDbCleanArchive.log, and is stored in the /usr/ecs/<mode>/CUSTOM/logs directory.

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.
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	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): *Level 0* - Reconstructed, unprocessed instrument/payload data at full resolution; any and all communications artifacts, e.g., synchronization frames, communications headers, duplicate data removed. *Level 1A* - 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. *Level 1B* - Level 1A data that have been processed to sensor units (not all instruments will have a Level 1B equivalent). *Level 2* - Derived geophysical variables at the same resolution and location as the Level 1 source data. *Level 3* - Variables mapped on uniform space-time grid scales, usually with some completeness and consistency. *Level 4* - Model output or results from analyses of lower level data (e.g., variables derived from multiple measurements).
Data Server	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.
DDTS	Change request manager used to compose, submit, report and track status of proposals to change ECS resources electronically.
ECS Assistant	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.
ECS Desktop	Simulates Common Desktop Environment (CDE); interface that acts like a file manager, allowing launch of applications, creation of directories and moving/copying/ deleting files.

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.
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.
FLEXlm	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 property 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.
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.

PlPRGenerator User Interface	The PlPRGenerator is the command line interface for the Production Request Editor. The PlPRGenerator 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 *EcDsSrUpdateESDTAccess* and *EcDsSrUpdateQATimeRange* provide DAAC operations staff the capability to adjust how the Science Data Server restricts *Acquire* access to granules. When evaluating a user's permission to *Acquire* a granule, the Science Data Server uses the value of the NASA user attribute stored in the User Profile system. The first script, *EcDsSrUpdateESDTAccess*, 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, *EcDsSrUpdateQATimeRange*, 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
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
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 Ssytem
EDC	EROS Data Center (DAAC)
EDF	ECS Development Facility
EDGRS	ESDIS Data Gathering and Reporting System
EDHS	ECS Data Handling System
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
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

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
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
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