S4PM 5.8.1 Operations Guide

A guide to operating NASA's open source Simple, Scalable, Script-Based, Science Processor for Measurements (S4PM)

October 2005

Document Version 1.0.0

Stephen W. Berrick, NASA

Table of Contents

1. INTRODUCTION 8

1.1 GOALS OF S4PM 8
1.2 FUTURE DIRECTIONS 9

2. RELATED DOCUMENTATION 10

3. FEATURES OF S4PM 11

3.1 S4PM CONFIGURATION OPTIONS 11
3.1.1 GETTING DATA 11
3.1.1.1 ECS SCLI Orders Via Subscription 11
3.1.1.2 ECS SCLI Order Without Subscription 11
3.1.1.3 ECS Pull Order Via Synergy 4 Path 12
3.1.1.4 Disk Source Via Polling 12
3.1.2 PUSHING DATA 12
3.1.2.1 ECS Archive Via PDR 12
3.1.2.2 ECS Datapool Insert 12
3.1.2.3 ECS Distribution 12

4. INSTALLING S4PM 13

4.1 INSTALLATION 13

5. RUNNING S4PM 14

5.1 THINGS YOU WILL NEED 14
5.2 ENVIRONMENT SETUP 14
5.2.1 S4P_LOG_LEVEL 14
5.2.2 OUTPUT_DEBUG 14
5.2.3 S4PM_CONFIGDIR 15
5.3 STARTING THE S4PM MONITOR 15
5.2.1 ANATOMY OF THE S4PM MONITOR 16
5.2.1.1 Title Bar 16
5.2.1.2 Station Buttons 17
5.2.1.3 Job Boxes 17
5.2.1.4 Success and Fail Tallies 17
5.2.1.5 Control Buttons 17
5.3 CONTROLLING STATIONS WITH STATION MONITOR 18
5.3.1 ANATOMY OF THE STATION MONITOR 20

5.3.1.1 Top Panel 20
5.3.1.2 Middle Panel 20
5.3.1.3 Bottom Panel 21
5.4 CONTROLLING JOBS WITH JOB MONITOR 21
5.5 RELATIONSHIP BETWEEN STATION MONITOR AND JOB MONITOR 23
5.6 S4PM ADMINISTRATION TOOL 23
5.7 ALGORITHM INFO TOOL 26

6. MONITORING S4PM 27

6.1 S4PM DRILL-DOWN 27
6.1.1 STATION DRILL-DOWN 27
6.1.2 JOB DRILL-DOWN 27
6.1.2.1 Drill-Down Into Running Jobs 28
6.1.2.2 Drill-Down Into Failed Jobs 28
6.1.2.3 Drill-Down The "Old Fashioned" Way 28
6.2 LOG FILES 30
6.2.1 STATION LOG FILES 32
6.2.2 CHAIN LOG FILES 32
6.2.3 TRANSACTION LOG FILES 35
6.2.3.1 Find Data Transaction Log File 35
6.2.3.2 Track Data Transaction Log File 36
6.3 DATABASE FILES 37
6.4 WORK ORDER FILES 37
6.4.1 WORK ORDER FILE NAMES 37
6.4.2 WORK ORDER FORMATS 38
6.4.2.1 PDR Format 38
6.4.2.2 PCF Format 40
6.4.2.3 ODL Format 40
6.4.2.4 Pathname List Format 40
6.4.2.5 E-Mail Format 41
6.5 DATA FILES 42
6.5.1 DATA FILE NAMES 42
6.5.2 METADATA FILES 44
6.5.3 UR FILES 44

7. WORKING WITH ALGORITHMS 46

7.1 WHAT ALGORITHMS CAN S4PM SUPPORT? 46
7.2 ALGORITHM PRODUCTION RULES 46
7.3 PRODUCTION RULE CONCEPTS 47
7.3.1 SIMPLE PRODUCTION SCENARIOS 47
7.3.2 THE STRINGMAKER ALGORITHM CONFIGURATION FILE 48
7.4 ALGORITHM INSTALLATION AND CONFIGURATION 48

8. THE ALLOCATE DISK STATION 49

8.1 STATION OVERVIEW 49
8.2 STATION WORK ORDERS 50
8.3 WHAT TO MONITOR 51
8.4 WHAT CAN GO WRONG 52

9. THE AUTO REQUEST STATION 53

9.1 STATION OVERVIEW 53
9.2 STATION WORK ORDERS 55

10. THE CONFIGURATOR STATION 56

10.1 STATION OVERVIEW 56
10.1.1 INSTALLING AN ALGORITHM 56
10.1.2 UNINSTALLING AN ALGORITHM 58
10.1.3 MODIFYING MAX JOBS 59
10.2 STATION WORK ORDERS 60
10.3 WHAT TO MONITOR 60
10.4 WHAT CAN GO WRONG? 60

11. THE EXPORT STATION 61

11.1 STATION OVERVIEW 61
11.2 STATION WORK ORDERS 62
11.3 WHAT TO MONITOR 62
11.4 WHAT CAN GO WRONG 62

12. THE FIND DATA STATION 63

12.1 STATION OVERVIEW 63
12.1.1 MANUAL OVERRIDES 64
12.2 STATION WORK ORDERS 66
12.3 WHAT TO MONITOR 66
12.3.1 SLEEP MESSAGE FILE 67
12.3.2 CHAIN LOG FILE 68
12.4 WHAT CAN GO WRONG 69
12.4.1 JOB STAYS GREEN SEEMINGLY FOREVER. 69
12.4.2 DATA FILE EXISTS, BUT FIND DATA CANNOT SEE IT. 69
12.4.3 JOB FAILED AND THEN THE REQUIRED DATA ARRIVES. 70
12.4.4 JOB SOMEHOW SUCCEEDED WITHOUT ALL THE REQUIRED DATA. 70

13. THE INSERT DATAPOOL STATION 71

13.1 STATION OVERVIEW 71
13.2 STATION WORK ORDERS 72

14. THE PREPARE RUN STATION 73

14.1 STATION OVERVIEW 73
14.1.1 MODIFY PCF RUNTIME PARAMETERS TOOL 73
14.2 STATION WORK ORDERS 76
14.3 WHAT TO MONITOR 76
14.4 WHAT CAN GO WRONG 76

15. THE POLL DATA STATION 78

15.1 STATION OVERVIEW 78
15.2 STATION WORK ORDERS 78

16. THE RECEIVE DN STATION 79

16.1 STATION OVERVIEW 79
16.2 STATION WORK ORDERS 80
16.3 WHAT TO MONITOR 80
16.4 WHAT CAN GO WRONG 80

17. THE RECEIVE PAN STATION 81

17.1 STATION OVERVIEW 81
17.2 STATION WORK ORDERS 82
17.3 WHAT TO MONITOR 83

18. THE REGISTER DATA/REGISTER LOCAL DATA STATIONS 84

18.1 STATION OVERVIEW 84
18.2 STATION WORK ORDERS 85
18.2.1 REGISTER DATA 85
18.2.2 REGISTER LOCAL DATA 85

19. THE REPEAT HOURLY/REPEAT DAILY STATIONS 86

19.1 STATION OVERVIEW 86
19.2 STATION WORK ORDERS 88
19.3 WHAT TO MONITOR 88

20. THE REQUEST DATA STATION 89

20.1 STATION OVERVIEW 89
20.1.1 COMPOSE REQUEST TOOL 89
20.2 STATION WORK ORDERS 91
20.3 WHAT TO MONITOR 91
20.4 WHAT CAN GO WRONG 92
20.4.1 DATA MULTIPLICITY 92
20.4.2 JOB FAILS 92
20.4.3 JOB STAYS BLUE IN THE QUEUE 92

21. THE RUN ALGORITHM STATION 94

21.1 STATION OVERVIEW 94
21.2 STATION WORK ORDERS 95
21.3 WHAT CAN GO WRONG 95

22. THE SELECT DATA STATION 96

22.1 STATION OVERVIEW 96
22.2 STATION WORK ORDERS 97

23. THE SHIP DATA STATION 98

23.1 STATION OVERVIEW 98
23.2 STATION WORK ORDERS 99

24. THE SPLIT SERVICES STATION 100

24.1 STATION OVERVIEW 100
24.2 STATION WORK ORDERS 101

25. THE STAGE FOR PICKUP STATION 102

25.1 STATION OVERVIEW 102
25.2 STATION WORK ORDERS 102

26. THE SUBSCRIPTION NOTIFY STATION 104

26.1 STATION OVERVIEW 104
26.2 STATION WORK ORDERS 105

27. THE SWEEP DATA STATION 106

27.1 STATION OVERVIEW 106
27.2 STATION WORK ORDERS 107

28. THE TRACK DATA STATION 108

28.1 STATION OVERVIEW 108
28.1.1 DELETE DATA TOOL 109
28.2 STATION WORK ORDERS 110
28.3 WHAT TO MONITOR 111
28.4 WHAT CAN GO WRONG 111

29. THE TRACK REQUESTS STATION 112

29.1 STATION OVERVIEW 112
29.2 STATION WORK ORDERS 113

APPENDIX A. ACRONYMS 114

1. Introduction

This document describes the operation of S4PM, version 5.8.1. For information on installing and configuring S4PM, please read the S4PM 5.8.1 Installation and Configuration Guide. This document assumes that S4PM has already been properly installed and configured.

The Simple, Scalable, Script-based Science Processor for Measurements (S4PM) is a NASA developed system for highly automated processing of science data. S4PM is the main processing engine at the Goddard Earth Sciences Data and Information Services Center (GES DISC). In addition to being scalable up to large processing systems such as the GES DISC, it is also scalable down to small, special-purpose processing strings.

S4PM consists of two main parts: the kernel is the Simple, Scalable, Script-based Science Processor (S4P), an engine, toolkit and graphical monitor for automating script-based, data-driven processing. The S4PM system is built on top of S4P and implements a fully functioning processing system that supports a variety of science processing algorithms and scenarios.

S4PM requires Perl (ideally 5.6 or higher) and has been run successfully on Irix, Linux (Red Hat), Solaris, Macintosh OS X, and Microsoft Windows.

S4PM was released to the open source community under the NASA Open Source Agreement in April 2005 with version 5.6.2. The software is available from SourceForge at this URL: http://sourceforge.net/projects/s4pm/.

1.1 Goals of S4PM

The main goal of S4PM is to automate science processing to the extent that a single operator can monitor all of the processing in an "industrial-size" data processing center. A second goal is to be flexible enough to easily add new processing strings or new algorithms to an existing string with a minimum of effort.

High usability is another key goal of S4PM, deriving from the need for more automation at less operational cost. Specific goals are:

- Allow a single operator to manage and monitor hundreds of jobs simultaneously.
- Drill down to troubleshoot a problem in two mouse clicks.

- Set up a new processing string in less than 30 minutes.

1.2 Future Directions

The architecture of S4PM and S4P was specifically designed to be highly modular so that it could evolve quickly and flexibly. It has already evolved from data-driven processing of MODIS instrument data to AIRS processing to on-demand subsetting based on user requests. Version 5.7.0 was the first release incorporating data mining into S4PM, allowing users to upload algorithms via a Web interface for execution at the GES DISC.

For the future, S4PM will evolve to:

- Support an ever-increasing variety of processing algorithms, scenarios and data interfaces.
- Increase the automation of failure monitoring and recovery.
- Reduce the time and expertise needed to setup and adapt S4PM to new processing algorithms.

We hope that some or all of these goals will be reached by collaborating with the open source community.

2. Related Documentation

The S4PM home page is at: http://disc.gsfc.nasa.gov/techlab/s4pm/ where the following documents are available:

- S4PM 5.8.1 Installation and Configuration Guide
- S4PM 5.8.1 Design Document
- S4PM 5.8.1 Release Notes

3. Features of S4PM

Table 3-1 below lists many of the features of S4PM.

Feature	Description
Portable	S4PM is portable to any system the runs Perl and Perl is one of the most ubiquitously supported development languages around.
Simple	What you see is what you get. Nothing is hidden. Related to this is transparency which means that there are no hidden layers (e.g. sockets). S4PM is based on a simple paradigm as well: an assembly line containing one or more stations that do work on one or more work orders which are then passed to other stations.
Easy	S4PM is designed to be easy to install, configure and get running. Other than familiarity with Perl, S4PM doesn't require any special software expertise to configure or even modify.

Table 3-1. Features of S4PM included in version 5.8.1.

3.1 S4PM Configuration Options

Owing to its flexibility, S4PM can be configured a number ways in how it gets data to process and what it does with the output products it generates.

3.1.1 Getting Data

S4PM can be configured to get data in one of several ways:

3.1.1.1 ECS SCLI Orders Via Subscription

In this configuration, subscriptions for notification of insert are set up in ECS for all external input data needed by S4PM. The receipt of an insert notification for particular data triggers the automatic ordering of those data via the Science Data Server Command Line Interface (SCLI) tool. The ordered data are then pushed to the input disk pool on the S4PM system. It is used mainly for near real-time processing of data.

3.1.1.2 ECS SCLI Order Without Subscription

In this configuration, data are ordered manually using the Compose Data Request tool, a GUI that communicates with the SCLI. Otherwise, it is similar to ECS SCLI Orders via Subscription described above. This configuration is more suited to ad-hoc or routine reprocessing.

3.1.1.3 ECS Pull Order via Synergy 4 Path

Release 5.7.1 of S4PM was the first to introduce the ability to configure a string for getting data via an FTP pull order from ECS Datapool. This assumes that the data types involved are configured for the Synergy 4 distribution path. When configured this way, symbolic links to the data are created in the S4PM input disk pool rather than the data physically getting moved there.

3.1.1.4 Disk Source via Polling

In this configuration, data reside on some disk external to S4PM. The assumption is that the external disk is the ECS Datapool or configured very similarly in terms of directory structure. S4PM polls the disk resource periodically for new data arrivals. As data show up, S4PM sets up temporary soft links to these data in the S4PM input disk pool.

3.1.2 Pushing Data

S4PM can be configured to push data in one of several ways:

3.1.2.1 ECS Archive via PDR

In this configuration, output data slated to be archived in the ECS are handled via PDR polling, the same mechanism defined for SIPS. Output from the running of each algorithm is described in a Product Delivery Record (PRD) which ECS periodically polls for in a designated directory. The data listed in the PDRs are then ingested into the ECS archive and a notification is sent back to S4PM.

3.1.2.2 ECS Datapool Insert

In this configuration, data are sent to the ECS Datapool bypassing the ECS archive. As with pushing to the ECS archive, the PDR mechanism is used to list the data that are to be pushed into the Datapool. A notification, similar to the one used above, is sent back to the S4PM upon completion.

3.1.2.3 ECS Distribution

This configuration is used for on-demand processing where data are produced on behalf of a user's specific request (e.g. for subsetting) via an ordering interface like the EOSDIS Data Gateway (EDG). In this case, the default mechanism is the External Product Dispatcher (EPD). The EPD, however, only supports requests coming from the V0 Data Gateway. Other requests (such as from WHOM, the GES DISC ordering interface) are handled via the ECS-provided Distribution Command Line Interface (DCLI).

4. Installing S4PM

4.1 Installation

For information on how to install S4PM, please refer to the S4PM 5.8.1 Installation and Configuration Guide.

5. Running S4PM

This section describes how to run S4PM. It assumes that S4PM has been correctly installed and configured on your system (see the S4PM 5.8.1 Installation and Configuration Guide).

5.1 Things You Will Need

In order to run S4PM, you will need these items:

1. The name of the machine or system where S4PM has been installed and configured.
2. The location of the S4PM root directory on that machine.
3. The userid and password of the account to be used for running and monitoring S4PM.

5.2 Environment Setup

There are three environment variables supported by S4PM and all are optional.

5.2.1 S4P_LOG_LEVEL

This environment variable sets the minimum severity of message to write into the S4PM chain log files. Valid values are FATAL, ERROR, WARNING, INFO, and DEBUG. The default is to write all messages to the chain logs (but see OUTPUT_DEBUG).

For example in C Shell:

```
setenv S4P_LOG_LEVEL ERROR
```

Or, in Korn, Bash, or Bourne shells:

```
export S4P_LOG_LEVEL=ERROR
```

5.2.2 OUTPUT_DEBUG

Note that OUTPUT_DEBUG is considered deprecated and S4P_LOG_LEVEL is the preferred way of controlling log messages. The OUTPUT_DEBUG environment variable enables or disables the writing of debug log messages to the chain log files. Set to a non-zero value to enable; set to zero or not set it at all to disable. Note that S4P_LOG_LEVEL trumps OUTPUT_DEBUG. The default is disabled.

For example in C Shell:

```
setenv OUTPUT_DEBUG 1
```

Or, in Korn, Bash, or Bourne shells:

```
export OUTPUT_DEBUG=1
```

5.2.3 S4PM_CONFIGDIR

This environment variable is only needed if S4PM is configured to use a different file naming convention than its default for data files. Data file names within S4PM can be modified with the $s4pm_filename_template parameter in the Stringmaker configuration files. When this option is used, the environment variable S4PM_CONFIGDIR must be set to the directory location of the Stringmaker configuration files.

Refer to the S4PM 5.8.1 Installation and Configuration Guide for more details.

5.3 Starting The S4PM Monitor

These are the steps for bring up the S4PM Monitor:

1. Log onto the system or machine where S4PM has been installed using your assigned userid and password.
2. Change directories to the root directory of S4PM on that machine:

   ```
   cd $S4PM_HOME
   ```

3. Start up the monitor:

   ```
   ./s4pm_start.ksh
   ```

   Within a few seconds, the S4PM Monitor graphical user interface should be displayed. It should look something like Figure 5-1 below.

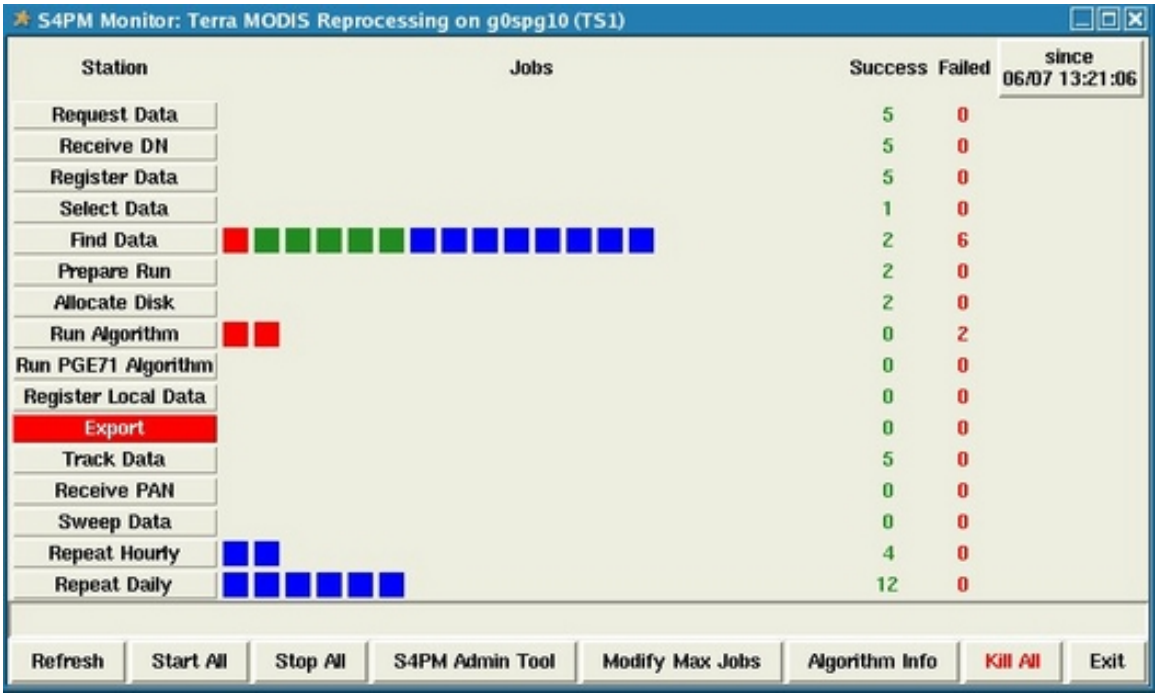

Figure 5-1. Example S4PM Monitor. Buttons along bottom are shortcuts to common tasks and tools. The Export station is red indicating that it is not running. The red job boxes in Run Algorithm and Find Data stations are failed jobs. The green boxes in Find Data station are running jobs and the blue ones are jobs that are queued up waiting to be run.

5.2.1 Anatomy Of The S4PM Monitor

The S4PM Monitor is the main interface for monitoring activity and status of a S4PM string. The S4PM Monitor updates the display on a configurable frequency. Figure 5-1 shows an example S4PM Monitor.

In the sections below, each of the aspects of the S4PM Monitor is discussed:

5.2.1.1 Title Bar

The title bar at the top of the window is labeled S4PM Monitor and also displays the name of the mission, the instance, and machine name. The mission name is a descriptive name for the data being processed. In this example, the data is from the MODIS Terra spacecraft platform. The instance refers to what instance or flavor of MODIS Terra; for each mission, there may be more than one instance such as forward processing and reprocessing. Finally, the machine (in this case, g0spg10) is displayed.

5.2.1.2 Station Buttons

The station buttons run down the left side of the window and represent stations configured to work in this S4PM string. If the station button is red (for example, the Export station in Figure 5-1), it means that station is not running. Otherwise, the station is running. Clicking on a station button brings up the Station Monitor window which will be discussed in Section 5.3.

Right-clicking on a station button is a shortcut to some of the things you can do in the Station Monitor such as stopping or starting the individual station. When you right-click on a station button, a popup menu allows you to select these options as well as any others configured for that station.

5.2.1.3 Job Boxes

Jobs running in each of the stations are represented with colored boxes to the right of the station button. Green boxes are running jobs; jobs waiting to be run (in the queue) are represented with blue boxes; and finally jobs that have failed are represented with red boxes. In Figure 5-1, the Find Data station (Section 12) has one failed job, five running jobs, and eight jobs that are in the queue to be run. The Run Algorithm station (Section 21) has two failed jobs. Finally, the Repeat Hourly and Repeat Daily stations (Section 19) have a number of jobs queued up waiting to be run.

Each station is configured for the maximum number of jobs that can be run at the same time. The default is five. Additional jobs that come into the station after the maximum is reached are queued up and are displayed as blue boxes. In Figure 5-1, there are eight jobs in the queue. Once a running jobs finishes or fails, one of the queued up blue jobs will then be run.

When a running job (green box) completes successfully, it disappears. If it fails, the box turns red and remains there until some action is taken. A red box indicates that something is wrong and should be tended to.

5.2.1.4 Success and Fail Tallies

On the right side of the window are two columns that maintain current tallies of jobs that have run successfully and those that have failed since the date and time listed on the **since** button. Clicking the **since** button resets all tallies back to zero.

5.2.1.5 Control Buttons

At the bottom of the window are a number of S4PM control buttons placed here for convenience. These are listed below in Table 5-1:

Button Name	Its Function
Refresh	Forces the S4PM Monitor to update its display.
Start All	Starts up all stations.
Stop All	Stops all stations. It does not affect any jobs in those stations that happen to be running.
S4PM Admin Tool	Brings up the S4PM Administration Tool interface which allows a number of mostly cleaning tasks to be accomplished.
Modify Max Jobs	Brings up the Modify Max Jobs Tool which allows adjustments to be made on the maximum number of jobs running in any one station.
Algorithm Info	Brings up the Algorithm Info window that lists for each algorithm registered, the algorithm name, version, input data types, and output data types.
Kill All	Stops all the stations, like the Stop All button, but also kills any jobs running in those stations.

Table 5-1. Buttons available in the S4PM Monitor window and their functions.

5.3 Controlling Stations With Station Monitor

Clicking on a station button in the S4PM Monitor (*e.g.* **Allocate Disk**) brings up another window. An example for the Allocate Disk station (Section 8) is shown below:

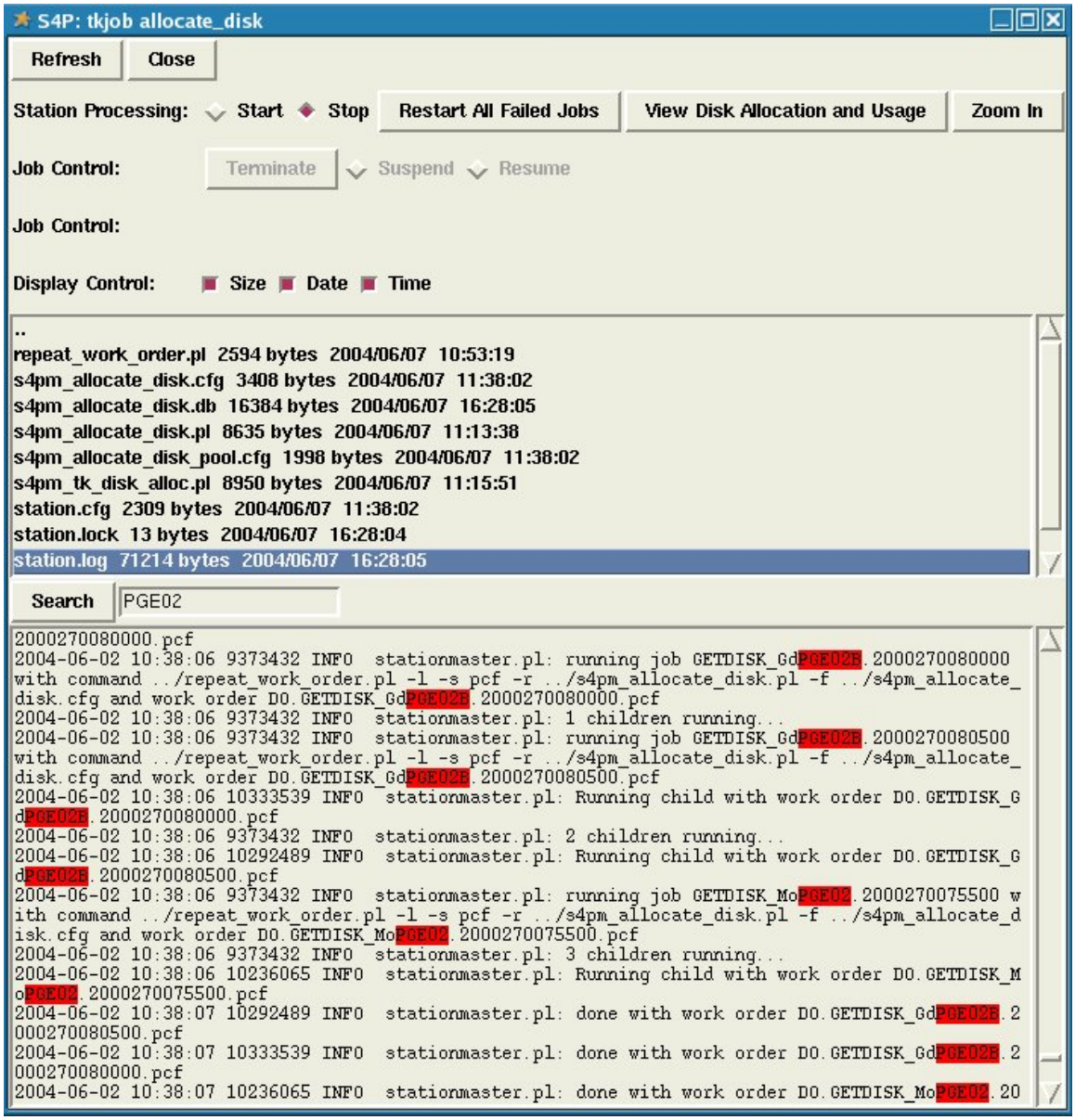

Figure 5-2. Example S4PM Station Monitor, in this case, for the Allocate Disk (or allocate_disk) station.

The S4PM Station Monitor is basically a view into the station's directory (remember, all stations are just directories): the top panel has some buttons and controls, the middle panel is a listing of the station directory's contents, and the bottom panel shows the contents of the file selected from the middle panel. In this example, the station is Allocate Disk (Section 8) and the actual station directory's name, allocate_disk, is shown in the title bar of the window. The contents of the file highlighted in the middle panel, station.log, are shown in the bottom panel. In addition, the search mechanism is also illustrated.

5.3.1 Anatomy Of The Station Monitor

The Station Monitor is the interface to a particular station running in S4PM. As with the main S4PM Monitor, the Station Monitor is updated periodically.

Below, each of the aspects of the Station Monitor is discussed:

5.3.1.1 Top Panel

The top panel of the Station Monitor contains buttons for controlling that station. Some buttons are station-specific, that is, they'll be seen in some stations, but not all. Several, however, are common to all stations. These are:

Button Name	Function
Refresh	To update the display now rather than waiting for the next polling cycle (default is 5 seconds)
Close	To close the Station Monitor window
Start and Stop Radio Buttons	To start or stop the station
Restart All Failed Jobs	To restart all failed jobs (red boxes) in the station.

Table 5-2. Station controls available in the Station Monitor window that are common across all S4PM stations.

In addition to these controls, a number of station-specific control buttons may be available to carry out tasks unique to those stations. In Figure 5-2, for example, the Allocate Disk station has the button **View Disk Allocation and Usage**. Clicking this button brings up the View Disk Allocation and Usage tool which shows a graphical representation of disk utilization. The **Zoom In** button starts up a new S4PM Monitor window focused in on the virtual substations making up the Allocate Disk station.

Note that the functionality of the **Start** and **Stop** Radio Buttons as well as the **View Disk Allocation and Usage, Zoom In,** and **Restart All Failed Jobs** buttons is also available by right-clicking on the station button in the S4PM Monitor window. Thus, you don't have to bring up a new window to get access to these controls.

Finally, the **Display Control** check boxes control the display of directory contents in the middle panel.

5.3.1.2 Middle Panel

The middle panel is a display of the contents in the station directory including other directories. Through this panel, you can navigate to other directories by double-clicking on the name. For example, double-clicking on .. (the two dots) will bring you up one directory.

A single click on a file will result in the contents of that file getting displayed in the bottom panel. Only ASCII text files can be displayed along with Perl DBM files (typically having a file name extension of .db). A single click on a directory will result in the contents of that directory getting displayed in the bottom panel. In this example, the user has clicked on the file station.log which is highlighted. The contents of that file are displayed in the bottom panel.

5.3.1.3 Bottom Panel

The bottom panel is used for displaying the contents of the file or directory highlighted in the middle panel. If nothing is selected in the middle panel, the bottom panel will be blank.

To help with diagnosing problems, any file name having the file name extension .log is assumed to be a log file and the display of that file will automatically go to the bottom where, most typically, the proximate cause of any problem can be found.

To facilitate troubleshooting, a rudimentary search button and entry box is provided. All matches will be highlighted in color in the file shown in the bottom panel and the display will jump to the first occurrence found. In this example, a search was done on the string "PGE02".

5.4 Controlling Jobs With Job Monitor

Clicking on a running job (green box) or a failed job (red box) brings up the Job Monitor window. The Job Monitor is essentially the same as the Station monitor, except that rather than showing you the contents of a station directory, the Job Monitor shows you the contents of a job directory (again, everything's just a directory). If the job is green and still running, the contents of the directory are likely to be continuously changing. If the job has failed, the contents are instead the debris left behind by the job's failure.

Figure 5-3 shows an example for a running job in the Find Data station. The Job Monitor was started by simply clicking on a green box in that station.

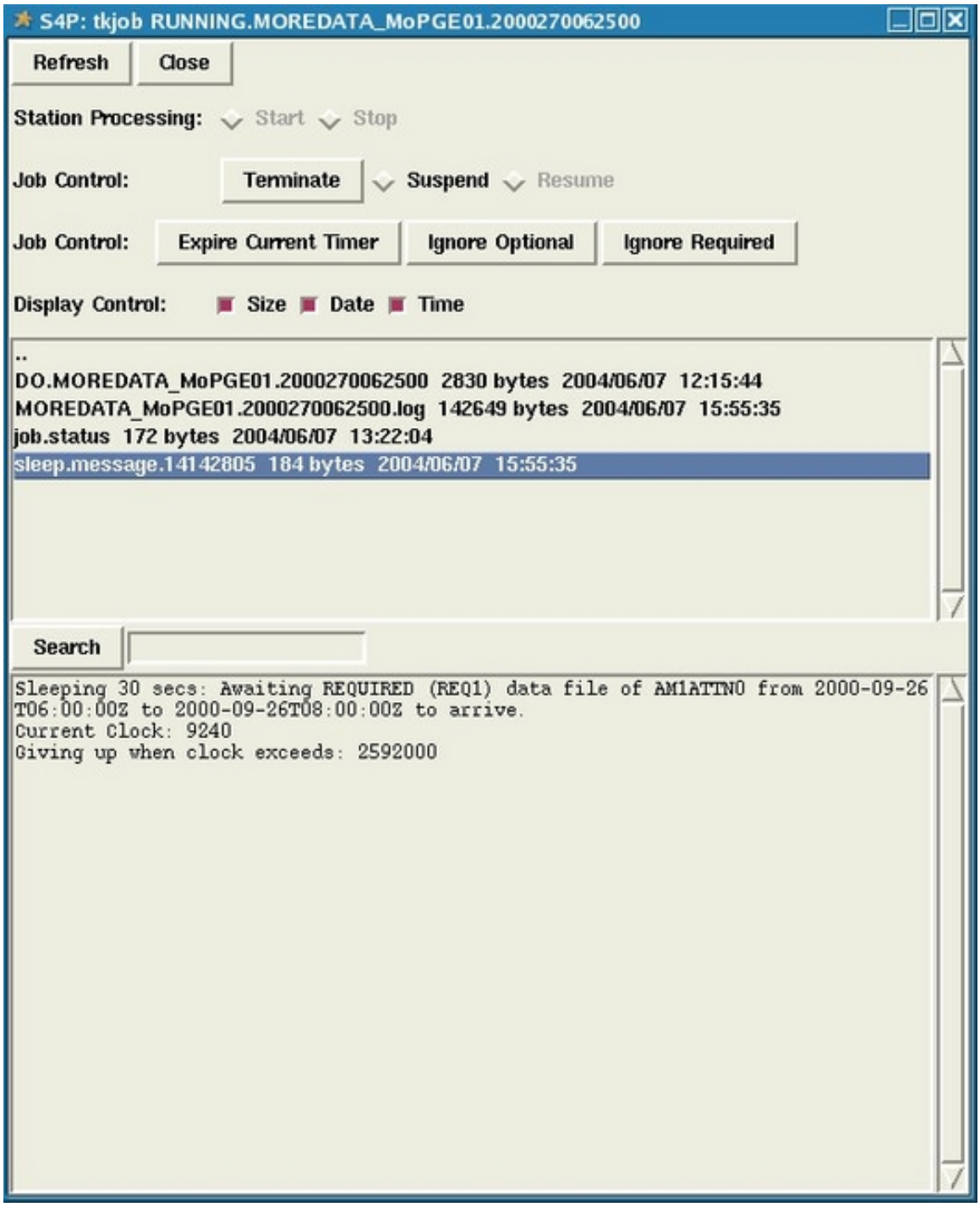

Figure 5-3. Example Job Monitor, in this case, for a running job in the Find Data station. Here, the sleep message file has been selected and its contents are shown in the bottom panel.

In Figure 5-3, the title bar contains the name of the running job directory, in this case: RUNNING.MOREDATA_MoPGE01.20022700062500. As with the Station Monitor, the top panel contains a number of buttons. But unlike with the Station Monitor, these

buttons are for controlling the job, not the station (you should notice that the **Start** and **Stop** radio buttons are grayed out, *i.e.* disabled). In the Find Data station (Section 12), the **Expire Current Timer**, **Ignore Optional**, and **Ignore Required** buttons cause changes to be made to the currently running job (the uses of these buttons is explained in Section 12).

As with the Station Monitor, the middle panel displays the contents of a directory, this time the job directory itself. The bottom panel displays the contents of any of the file or subdirectory in that directory. In this case, the contents of the file sleep.message.14142805 are shown. In the Find Data station, this is a continuously updated file; clicking on the same file name again shows the updated contents of the file.

5.5 Relationship Between Station Monitor and Job Monitor

As indicated above, the Station Monitor and Job Monitor are essentially the same window. The only distinction is the directory being viewed. If the directory is a station directory, the interface is a Station Monitor; if the directory is a job directory within a station directory, the interface is a Job Monitor. You can verify this yourself by bringing up the Job Monitor on some running or failed job, then navigating in the middle panel to one directory up (double click on the ..) into the station directory. Viola! The interface will change from being a Job Monitor to a Station Monitor. The control buttons in the top panel will even change appropriately. Furthermore, you can navigate back down to another station directory. Again, the control buttons that show up will be those for that new station (some may disappear, others may show up anew).

5.6 S4PM Administration Tool

Clicking on the **S4PM Admin Tool** button in the S4PM Monitor starts up the S4PM Administration Tool. This tool allows users to run a number of administrative tasks in S4PM. The Table 5-2 below lists that tasks that can be accomplished via the S4PM Administration Tool.

Task	Function
Clean Station Log Files	Remove all station.log and station_counter.log files from all station directories except Subscription Notify (Section 26).
Clean Other Log Files	Remove the find_data.log and cbr.log files after backing up compressed copies of them.
Clean Failed Jobs	Remove all failed jobs from all stations except Subscription Notify
Reset Databases	Reset the Track Data (Section 28) and Allocate Disk (Section 8) databases and delete the Allocate Disk transaction log file. In addition, clean out the disk pools of all data.
Locate unknown files and optionally delete them	Locate and optionally delete any unrecognized files from all station directories. A confirmation is required for each deletion.
Clean Out Archived Files	Clean out all files under the ARCHIVE directory.
Clean Out PH Files	Clean out all Production History (PH) files in the PH subdirectory under DATA.
Clean Out Blocks	Clean out all time-based blocks in the Register Data and Register Local Data stations (Section 18).
Full Clean Out	This essentially returns the S4PM string to its pristine state when it was first deployed.

Table 5-2. Tasks available via the S4PM Administration Tool.

Figure 5-4 below shows the S4PM Administration Tool main window along with the pop-up box that results from the successful completion of the tasked that have been selected.

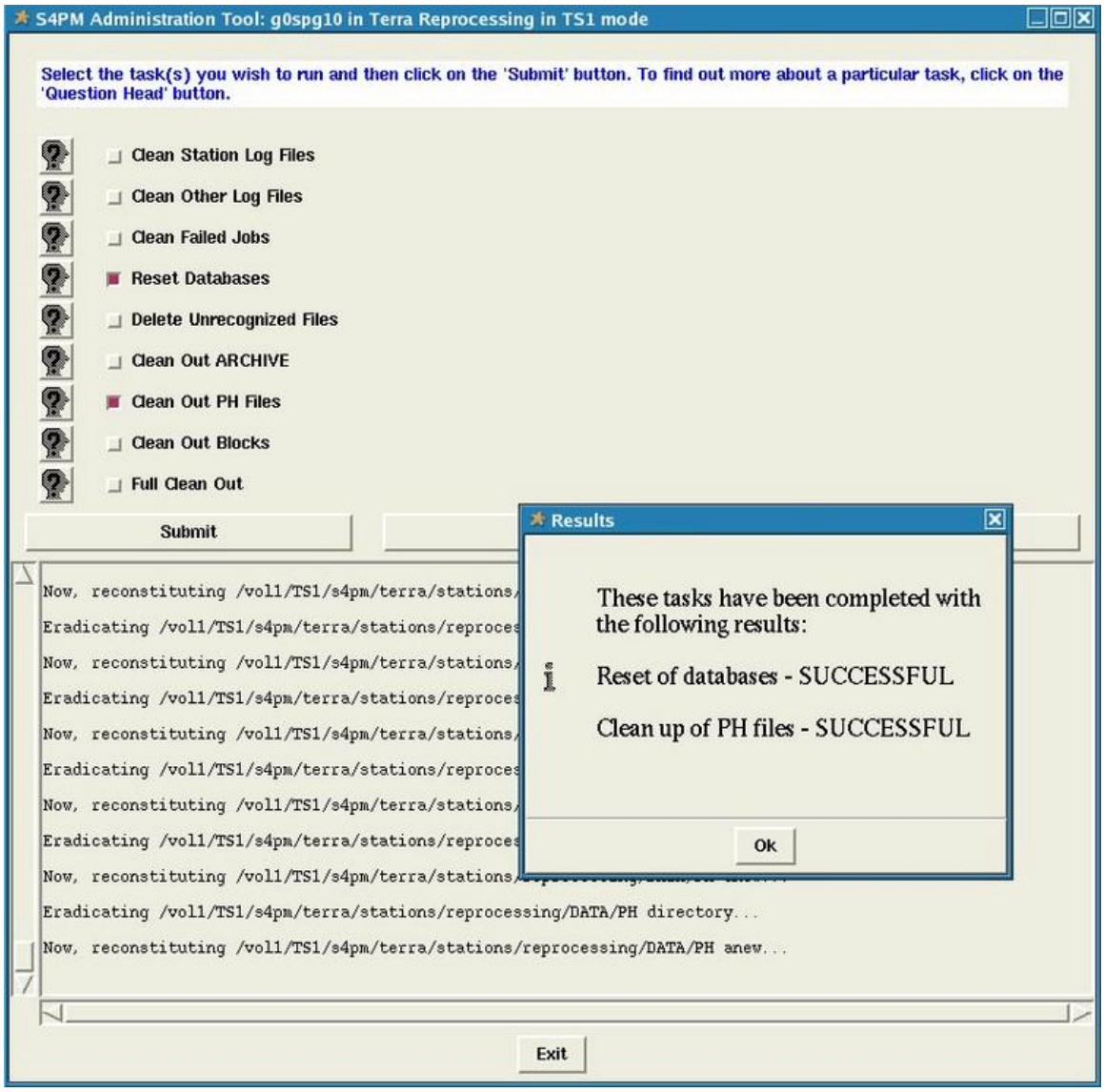

Figure 5-4. Example session with the S4PM Administration Tool. In this example, the user has selected two tasks: Reset Databases and Clean Out PH Files, both of which completed successfully as indicated by the pop-up box. Click on image to zoom in.

The S4PM Administration Tool can also be started up from the command line. First, you ***must*** be in the station root directory, the directory under which are all of the individual station subdirectories. The command to start the S4PM Administration Tool is:

```
s4pm_tk_admin.pl
```

S4PM Administration Tool can also be run in a terminal window without bringing up a graphical user interface (which improves performance). To do so, the command is:

`s4pm_tk_admin.pl -task` [Taskname]

where [Taskname] is one of :

Taskname	Is equivalent to this in the GUI …
FullCleanOut	Full Clean Out
Archive	Clean Out ARCHIVE
PH	Clean Out PH Files
Blocks	Clean Out Blocks
ExtraFiles	Delete Unrecognized Files
FailedJobs	Clean Failed Jobs
ResetDatabases	Reset Databases
StationLogFiles	Clean Station Log Files
OtherLogFiles	Clean Other Log Files

Table 5-3. Valid task names for running the S4PM Administration tool in a non-GUI mode and what they are equivalent to in the GUI version.

For example:

```
s4pm_tk_admin.pl -task FullCleanOut
```

is equivalent to bring up the S4PM Administration Tool window and selecting **Full Clean Out**.

5.7 Algorithm Info Tool

This simple tool is available by clicking on the **Algorithm Info** button in the S4PM Monitor. The Algorithm Info Tool simply displays the names of the algorithms currently registered in this S4PM string, their versions, input data types, and output data types. An example image of the display is shown below in Figure 5-5.

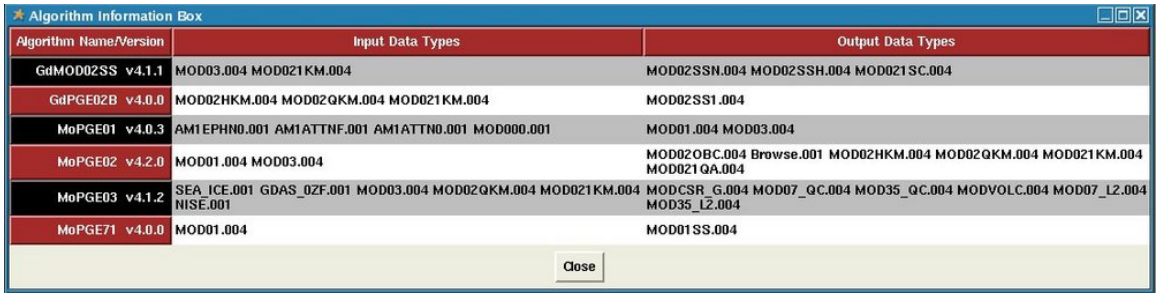

Figure 5-5. Example display of the Algorithm Info Tool window. Algorithm names and versions are listed down the left side. For each, the input and output data types are shown. Click on image to zoom in.

6. Monitoring S4PM

This section describes the tools and features S4PM that allow you to examine S4PM as it is running and diagnose problems when they occur. In the subsequent sections on individual stations, techniques for applying these tools and features to particular troubleshooting efforts are discussed.

This section assumes that S4PM has been correctly installed and configured on your system. It further assumes that you are familiar with the basic S4PM interfaces discussed in Section 5.

6.1 S4PM Drill-Down

S4PM was designed from the start to be easy to drill down and find sources of problems when they occur.

6.1.1 Station Drill-Down

Stations are just directories. The simplest way to drill down into a station and view its contents is to click on the station button (for example, Allocate Disk) in the S4PM Monitor. As discussed in Section 5.3, this results in the Station Monitor which presents a navigable view of the station directory. To view the contents of a file in that directory (assuming it is ASCII or a Perl DB file), just click on that file. Since S4PM is designed so that everything used by a station is visible in that station's directory, this tool allows all the stations parts to be examined in gory detail.

Particularly interesting and often useful files to examine in a station directory are the station.log file (Section 6.2.1) and the station.cfg file, the latter being the configuration file that dictates many properties of that station. Other interesting files include:

- In the Find Data station, the find_data.log transaction log file (Section 6.2.3.1) is available and often useful.
- In the Track Data station, the transaction.log file (Section 6.2.3.2) contains a list of all database transactions made in that station. In addition, the uses.db database file lists data files currently within S4PM and their uses and the path.db shows how these same data files are mapped to disk pools.
- In the Allocate Disk station, s4pm_allocate_disk.db is a database file showing the current state of the S4PM disk pools.

6.1.2 Job Drill-Down

As discussed in Section 5.4, clicking on a running green job box or a failed red job box brings up the Job Monitor. This is a view into the contents of a particular job directory.

6.1.2.1 Drill-Down Into Running Jobs

Running jobs are not typically drilled down into on a routine basis, but there are a few exceptions:

- In the Find Data station, jobs stay green a long time sometimes waiting for required or optional input to arrive. While a job is waiting on data, a sleep.message.nnnnnnn file exists and is updated periodically to convey what data the job is currently waiting on. Click on this file to see this information.
- In the Run Algorithm station, it may be useful to examine the LogStatus file. This is not a S4PM log file, but instead an ECS Toolkit log file (algorithms not using the ECS Toolkit will not produce this file). The format and contents are not S4PM "standard", but are instead set by the algorithm.

6.1.2.2 Drill-Down Into Failed Jobs

More typically, drill down is most effective for looking into why a job failed. With one click, you are taken into the failed job directory where you can examine log and other files. If the problem can be fixed, you may be able to click on **Restart** in the Job Monitor to restart the job.

6.1.2.3 Drill-Down The "Old Fashioned" Way

Not to be overlooked, you can always open up an xterm and examine any S4PM directory with simple UNIX commands such as cd, ls, more, vi, etc. (or if in Windows, their equivalents).

All S4PM stations are directories under some root directory which represents the top of the S4PM station tree for that S4PM string. Directory names for stations should be recognizable:

Formal Station Name	Directory Name
Allocate Disk	allocate_disk
Auto Request	auto_request
Configurator	configurator
Export	export
Find Data	find_data
Prepare Run	prepare_run
Receive DN	receive_dn
Receive PAN	receive_pan
Register Data	register_data
Register Local Data	register_local_data
Repeat Daily	repeat_daily
Repeat Hourly	repeat_hourly
Request Data	request_data
Run Algorithm	run_algorithm
Select Data	select_data
Ship Data	ship_data
Split Services	split_services
Stage For Pickup	stage_for_pickup
Subscription Notify	sub_notify
Sweep Data	sweep_data
Track Data	track_data
Track Requests	track_requests

Table 6-1. Station names and their associated directory names in S4PM.

If you change directories into a station directory (*i.e.* with cd), you will see the same contents as you would have if you clicked on that station's button in the S4PM Monitor.

Jobs in a station are also directories; they are subdirectories within the station directory. Thus, jobs running in the Find Data station are all subdirectories under the find_data directory.

A running job is a directory always starts with RUNNING. and is represented as a green box in the S4PM Monitor. Failed jobs are in directories that start with FAILED.* and are represented as red in the S4PM Monitor. Jobs that are still in the queue to be run (blue boxes in the S4PM Monitor) are not directories at all yet, but instead are work order files. These usually have named like DO.*.wo, but not always.

The following examples illustrate S4PM directory naming:

A job directory with this name:

find_data/RUNNING.MOREDATA_MoPGE03.2004057024000/

means this:

This is a job running in the Find Data station. The work order type is MOREDATA_MoPGE03 and it is specifically for an algorithm named MoPGE03. It would show up as green on the S4PM Monitor.

A job directory with this name:

run_algorithm/RUNNING.RUN_MoPGE02.2004057035000/

means this:

This is a job running in the Run Algorithm station, this time for MoPGE02. It would also be represented with a green box in the S4PM Monitor.

A job directory with this name:

run_algorithm/FAILED.RUN_IMAPPSST.2000270064000.29328960/

means this:

This is a job that has failed in the Run Algorithm station for another algorithm named IMAPPSST. It would show up as red on the S4PM Monitor.

A job directory with this name:

alloc_disk/DO.GETDISK_GdMOD02SS.2004056180500.pcf

means this:

This is not a directory, but a file in the Allocate Disk station. It is, in fact, a work order waiting to be processed, that is, it's in the queue to be run on behalf of an algorithm named GdMOD02SS. In the S4PM Monitor, it would show up as a blue box.

6.2 Log Files

All stations produce log files while they are running. Log files are among the first places to look at when something fails. Several characteristics are common to all S4PM log files:

1. All stations have at least one log file named station.log in the station directory.

2. Most station jobs produce log messages that are appended to a "chain" log, a log file that moves from station to station along with its companion work order.
3. All log files have the .log file name extension.
4. All log files are ASCII text files and are intended to be readable and convey clear information, particularly in the event of a problem.
5. When a log file is selected in the Station Monitor or Job Monitor, the display automatically jumps to the bottom of the file, the location where you most likely are to find the culprit of a problem.
6. Almost all the entries in a log file have a standard format: date stamp, time stamp, process ID of the program generating the entry, the severity mnemonic (see Table 6-1), the program name, and then finally the message itself. See Figure 6-1 for an example.

```
2004-02-06 14:16:32 16723248 INFO  s4pm_select_data.pl: **********
s4pm_select_data.pl starting algorithm MoPGE01 and work order
DO.NEWDATA_MoPGE01.2004037120000 **********
```

Figure 6-1. Sample log file entry showing the standard format. This log message was written on February 6, 2004 at 14:16:32 by the program s4pm_select_data.pl which is run in the Select Data station. The job ID is 16723248, a unique identifier of the particular job. The message is informational only (INFO) and only conveys that the program is starting up on a work order named DO.NEWDATA_MoPGE01.2004037120000.

Each message in the log file is time stamped and contains a severity mnemonic indicating the severity of the message:

Severity Mnemonic	Meaning
DEBUG	Only appears when debugging is enabled and is used solely for debugging purposes (debugging can be enabled by setting the environment variable OUTPUT_DEBUG to a non-zero value).
INFO	The message is for informational purposes only.
WARNING	A minor non-fatal warning condition was detected. These aren't generally serious except when they're followed soon by a failure.
ERROR	A major non-fatal error condition was detected. These are significant and often presage a fatal condition.
FATAL	A fatal error condition was detected and this results in a job failure.

Table 6-2. Standard S4PM error severity mnemonics and their meanings.

The minimum severity of messages written to the chain log files can be controlled via the S4P_LOG_LEVEL environment variable. Valids are FATAL, ERROR, WARNING, INFO, and DEBUG.

6.2.1 Station Log Files

As indicated above, all stations have the station.log file. This log file simply contains a record of all work orders that have passed through the station: the name of the work order received and how it was dispositioned. As in all log files, each time stamped message is associated with a severity mnemonic as listed in Table 6-2.

6.2.2 Chain Log Files

Chain logs are log files that accumulate messages as a work order moves from station to station. In effect, they are the concatenation of all log messages associated with a single thread moving through S4PM from receipt of the data via the Register Data station through production of the downstream products.

Unlike the station.log files, however, chain log files exist in the job directories, not the station directories. Further, chain logs do not have a consistent file name. Rather, the file name changes as the log travels from station to station with its companion work order. In fact, the file name of the chain log is the same as the name of the work order except that (1) there is no DO. in front, and (2) the file name extension .wo is replaced by .log.

For example, a work order sent to the Select Data station might have the name:

DO.NEWDATA_GdPGE02B.2004020130500.wo

The chain log associated with that work order would have the name

NEWDATA_GdPGE02B.2004020130500.log

When Select Data completes processing this work order, it sends a new work order to Find Data named

DO.MOREDATA_GdPGE02B.2004020130500.wo

The chain log from Select Data is renamed to

MOREDATA_GdPGE02B.2004020130500.log

and also sent to the Find Data station. It is important to note that the MOREDATA_GdPGE02B.2004020130500.log contains everything that NEWDATA_GdPGE02B.2004020130500.log contained, but new messages from the Find Data job will be "chained" to the bottom. This process of renaming log files and

Note: Only the logs are chained, work orders are not. Unlike a chain log, a work order does not retain heritage information.

adding new messages continues as work order and log thread their way through S4PM.

In addition to containing a concatenation of all log messages from each station it passed through, the chain log also contains the contents of the input work order. In this way, the information contained in the input work orders is preserved in the chain log files. If you examine a chain log, you will see the information laid out in the following way:

1. A line of equal signs used as a separator.
2. A line containing a date and timestamp, process ID, and station name.
3. The contents of the work order that was sent into that station.
4. One or more log messages produced by that station.
5. Another line of equal signs used as a separator.
6. And then the same information is repeated for the next station the work order was sent to, etc.

As an example, an excerpt of a chain log as it went from the Select Data station to the Find Data station is shown in Figure 6-2. The file name of this chain log would have changed as described above.

```
==============================================================
2004-02-06 14:16:31 16280853 Select Data
TOTAL_FILE_COUNT=2;
OBJECT=FILE_GROUP;
        DATA_TYPE=MOD000;
        DATA_VERSION=001;

UR=UR:10:DsShESDTUR:UR:15:DsShSciServerUR:13:[GSF:DSSDSRV]:22:SC:MOD000.
001:39996855;
        DATA_START=2004-02-06T12:00:00Z;
        DATA_END=2004-02-06T13:59:59Z;
        OBJECT=FILE_SPEC;

DIRECTORY_ID=/CUSTOM/terra/forward/DATA/INPUT/MOD00.A2004037.1200.001.20
04037173144;
                FILE_TYPE=SCIENCE;
                FILE_SIZE=1999164888;
                FILE_ID=P0420064AAAAAAAAAAAAAA04037172142001.PDS;
        END_OBJECT=FILE_SPEC;
        OBJECT=FILE_SPEC;

DIRECTORY_ID=/CUSTOM/terra/forward/DATA/INPUT/MOD00.A2004037.1200.001.20
04037173144;
                FILE_TYPE=SCIENCE;
                FILE_SIZE=1999978282;
                FILE_ID=P0420064AAAAAAAAAAAAAA04037172142002.PDS;
        END_OBJECT=FILE_SPEC;
END_OBJECT=FILE_GROUP;
----------------------------------------------------------------------
2004-02-06 14:16:32 16723248 INFO  s4pm_select_data.pl: **********
s4pm_select_data.pl starting PGE MoPGE01 and work order
DO.NEWDATA_MoPGE01.2004037120000 **********
2004-02-06 14:16:32 16723248 INFO  s4pm_select_data.pl: **********
s4pm_select_data.pl completed successfully! **********
==============================================================
2004-02-06 14:16:33 16718697 Find Data
TOTAL_FILE_COUNT=13;
PROCESSING_START=2004-02-06T11:55:00Z;
PROCESSING_STOP=2004-02-06T12:10:00Z;
POST_PROCESSING_OFFSET=-300;
PRE_PROCESSING_OFFSET=0;
OBJECT=FILE_GROUP;
```

```
        DATA_TYPE=MOD000;
        DATA_VERSION=001;
UR=UR:10:DsShESDTUR:UR:15:DsShSciServerUR:13:[GSF:DSSDSRV]:22:SC:MOD000.
001:39996855;
        NEED=REQ1;
        LUN=599002;
        DATA_START=2004-02-06T12:00:00Z;
        DATA_END=2004-02-06T13:59:59Z;
        TIMER=0;
```

Figure 6-2. Excerpt from a chain log as it went from the Select Data station to the Find Data station.

The chain logs are one of the more valuable assets for troubleshooting problems since they preserve all work orders and all messages for a S4PM thread in a single file.

Note that not all messages from a station are preserved in chain logs. Due to branching (one station sending work orders to more than one downstream station), some information from “minor” branches is lost.

6.2.3 Transaction Log Files

In addition to station logs and chain logs, several S4PM stations maintain information in transaction logs, log files that record a transactional activity in the station. Transaction log files differ from station log files and chain log files in that they do not adhere to the format described in Section 6.2. Like the station log files, transaction log files reside in the station directory.

6.2.3.1 Find Data Transaction Log File

The Find Data transaction log file is named find_data.log and it resides in the Find Data station directory. This log file records information on what data Find Data was able to locate for a particular algorithm. S4PM production rules allow much freedom in how an algorithm expresses the recipe for making a product and this includes allowing the algorithm to have one or more choices depending upon what data are available at any given time. A recipe might call for the current day's NISE data if available, but if it's not available, then use the previous day's NISE data instead. Find Data ultimately resolves what is and isn't available for every run. The find_data.log records what Find Data actually found. This information is recorded in the log file in a compact form for every job passing through the station.

```
PGE=MoPGE01 DATADATE=2000-09-26T07:25:00Z PRODDATE=2004-02-24T15:50:21
MOD000=CURR AM1EPHN0=CURR AM1ATTN0=CURR

PGE=MoPGE01 DATADATE=2000-09-26T07:40:00Z PRODDATE=2004-02-24T15:50:21
MOD000=CURR AM1EPHN0=CURR AM1ATTN0=CURR

PGE=MoPGE03 DATADATE=2003-12-29T19:30:00Z PRODDATE=2003-12-30T11:54:59
MOD021KM=CURR MOD03=CURR MOD02QKM=NONE GDAS_0ZF=CURR NISE=CURR
SEA_ICE=PREV1
```

Figure 6-3. Excerpt from a Find Data find_data.log file.

Figure 6-3 shows an excerpt from the Find Data transaction log. The format is a single line of parameter=value pairs listing the algorithm name, data date/time, and production date/time. Following this are pairs of input data type and "currency". Currency describes how the time coverage of the data relates to the processing period of the algorithm: CURR means current, PREVn means previous, and FOLLn means following where n is an integer indicating how far back or how far forward the input is.

In this example, the last line is for algorithm MoPGE03. The particular run of this algorithm used the current MOD021KM and current MOD03 as inputs. Evidently, the MOD02QKM wasn't available so it used NONE (it is an optional input for MoPGE03). The GDAS_0ZF and NISE inputs were also current. But the SEA_ICE is not. It instead used the previous one, that is, for the previous day. PREV2 would've meant the day before that.

6.2.3.2 Track Data Transaction Log File

The Track Data station maintains a transaction log file named transaction.log in the station directory. This is more of a "pure" transaction log in that it actually records transactions made in the database of the Track Data station. All work orders processed by the Track Data station are recorded by a single entry in the log file. Figure 6-4 shows an excerpt.

```
2003-12-30 10:56:43 INSERT
/usr/ecs/OPS/CUSTOM/g0spg10/s4ins/terra/forward/

DATA/INPUT/AM1EPHN0.A2003363.2200.001.2003364053922 9

2003-12-30 11:48:27 UPDATE
/usr/ecs/OPS/CUSTOM/g0spg10/s4ins/terra/forward/

DATA/INPUT/MOD000.A2003363.1800.001.2003364001705 -1
UPDATE_MoPGE01_2003363194000

2003-12-30 12:00:50 DELETE
/usr/ecs/OPS/CUSTOM/g0spg10/s4ins/terra/forward/

DATA/MOD02SSH/MOD02SSH.A2003363.1930.004.2003364165511.hdf
```

Figure 6-4. Excerpt from a Track Data tranaction.log file.

As seen in Figure 6-4, each entry is time stamped. Next, the type of transaction is recorded. Possible types include INSERT (when a data file is inserted into the database),

UPDATE (to update the uses in the database), DELETE (when a data file is deleted from the database), and EXPECT (when a data file is expected). Following the transaction type is the full pathname of the data file itself followed by the number of uses.

6.3 Database Files

S4PM uses several databases to track granules and available disk space. These databases are Perl DB files that use the file name extension .db. Table 6-3 summarizes the stations that make use of these databases, the database file names, and their contents.

Station	Databases	What They Contain
Allocate Disk	s4pm_allocate_disk.db	All disk pool pathnames and the current space available in each.
Track Data	path.db	Current granules within S4PM and the disk pool in which they reside.
Track Data	uses.db	Current granules within S4PM and the number of uses currently remaining.

Table 6-3. Databases used within S4PM and what they contain.

As stated above, database or *.db files are in Perl DB format. As such, they are not ASCII files and cannot be viewed with simple commands like more or vi. There are, however, two simple ways to view these files:

1. If you select a DB file in the Station Monitor or Job Monitor, you can view the contents in the bottom panel of the display. These monitors convert the DB format into a readable form automatically.
2. On the command line, you can use the script dbls.pl as in: dbls.pl s4pm_allocate_disk.db

6.4 Work Order Files

As discussed in Section 4, work orders define what work gets done in a station and how. All work orders are ASCII files and hence, can be viewed via the Station Monitor, the Job Monitor, or via simple UNIX or Windows commands like more and vi. When a recognized work order shows up in a station, the Stationmaster of that station processes that work order by creating a subdirectory under which to execute the station code on the work order. Because work orders contain the information that triggers the processing in a station, it may be useful at times to view a work order's contents.

6.4.1 Work Order File Names

Work order file names generally adhere to a specific format:

DO.jobtype.jobid.wo

The jobtype identifies the type of work order it is. Stations are configured to only recognize the job types that they are responsible for processing. The jobid is unique to a particular work order and sets one work order apart from all others, even those of the same job type. The following are example work orders:

1. `DO.EXPORT_PH.MoPGE71PH_2004057095000.wo`
2. `DO.CLEAN_FAILPGE.20074867.wo`
3. `DO.MOREDATA_GdPGE02C.2004056175500.wo`

In the first example, the job type is EXPORT_PH and the job ID is MoPGE71PH_2004057095000.

In the second example the job type is CLEAN_FAILPGE and the job ID is 20074867.

In the third it is MOREDATA_GdPGE02C and 2004056175500.

As hinted above, deviation from this standard work order naming convention is allowed via configuration parameters. The work orders for some stations, for instance, use the file name extension .pcf rather than the standard .wo. This happens to be the case in the Allocate Disk and Run Algorithm stations. In other cases, the work orders do not have the DO. in front. Such is the case for the Receive DN station. In all cases, the S4PM Monitor understands these deviations in work order file names and represents them appropriately as blue boxes.

6.4.2 Work Order Formats

Work orders generally adhere to one of several formats depending on the station. The formats will be discussed briefly here. The work order formats in S4PM are:

- PDR - Product Delivery Record
- PCF - Process Control File
- ODL - Object Data Language
- Pathname List - List of full pathnames
- E-Mail - Varied

6.4.2.1 PDR Format

PDR or Product Delivery Record format is based in the format developed for EOSDIS for data providers to send and archive data in the ECS. The format actually has its heritage in ODL or Object Data Language.

PDR is an object oriented format. The container objects are FILE_GROUP and FILE_SPEC (within a FILE_GROUP). In addition to the PDR format as defined for EOSDIS, S4PM has added several other attributes to support stations within S4PM. These are described in Table 6-4 below:

Container	New Attribute	Description
PDR	PROCESSING_START	Algorithm processing start time and date.
	PROCESSING_STOP	Algorithm processing stop time and date.
	PRE_PROCESSING_OFFSET	Offset to be applied to the processing time before that time is used to compute the times of other data needed.
	POST_PROCESSING_OFFSET	Offset to be applied to the processing time only after that time is used to compute the times of other data needed.
FILE_GROUP	UR	UR is either the UR retrieved from the ECS Science Data Server database or, for data produced locally, the granule's local granuld ID (LGID).
	NEED	Input files can be required (REQn) or optional (OPTn) where n represents the order of preference with 1 being the most desired, 2 being next, etc.
	LUN	LUN refers to the logical unit number for that input granule in the process control file used during the running of the algorithm.
	DATA_START	Start date and time of the input trigger data.
	DATA_END	End date and time of the input trigger data.
	TIMER	The TIMER is the number of seconds to wait for that data before giving up on it. For required input, the timer starts once the trigger data granule arrives; for optional input, the timer starts once all of the required data granules have arrived.
	CURRENCY	The CURRENCY indicates the temporal coverage of the data relative to the processing period over which the algorithm is to be run. The processing period is aligned to the temporal coverage of the trigger data granule. CURRENCY has values like CURR (for current), PREVn (for previous), and FOLLn (for following) where n is an integer indicating how far away from the processing period it must be. Thus, PREV1 is the granule just prior to the current processing period, PREV2 is the one just before that, etc.

	BOUNDARY	BOUNDARY sets a reference in time against which processing periods of algorithms are computed. Example values are START_OF_DAY, START_OF_HOUR, START_OF_WEEK.

Table 6-4. Additional PDR attributes used by S4PM.

Stations whose input work orders are in PDR format are:

- Export
- Find Data
- Register Local Data
- Prepare Run
- Register Data
- Request Data
- Select Data

6.4.2.2 PCF Format

The Process Control File or PCF work order format is used by the Run Algorithm and Allocate Disk stations. This format is based upon the PCF format used by the ECS Science Data Processing Toolkit. It is basically a list of entries that map numerical identifiers (called logical unit numbers or LUNs) to physical files and directories. It is also used to map LUNs to runtime parameters that can serve as input to an algorithm.

It is a RUN work order sent by Allocate Disk the Run Algorithm station.

Stations whose input work orders are in PCF format are:

- Allocate Disk
- Run Algorithm

6.4.2.3 ODL Format

The ODL is object data language format. ODL is actually the format adopted and tailored for PDRs.

Stations whose input work orders are in ODL format are:

- Split Services

6.4.2.4 Pathname List Format

The pathname list format is a generally a list of full pathnames of files to be processed. For example, pathname list work orders are sent to the Sweep Data station and merely list out the full pathnames of files that can be deleted. Work orders sent to the Track Data

station are similar but also include the number of uses for each file. See Figures 6-5 and 6-6 below:

```
/usr/ecs/OPS/CUSTOM/g0spg10/s4ins/terra/forward/DATA/MODCSR_G/
MODCSR_G.A2003331.0610.004.2003331144742.hdf
/usr/ecs/OPS/CUSTOM/g0spg10/s4ins/terra/forward/DATA/MOD35_QC/
MOD35_QC.A2003331.0610.004.2003331144742.hdf
/usr/ecs/OPS/CUSTOM/g0spg10/s4ins/terra/forward/DATA/MOD07_L2/
MOD07_L2.A2003331.0610.004.2003331144742.hdf
/usr/ecs/OPS/CUSTOM/g0spg10/s4ins/terra/forward/DATA/MODVOLC/
MODVOLC.A2003331.0610.004.2003331144742.hdf
/usr/ecs/OPS/CUSTOM/g0spg10/s4ins/terra/forward/DATA/MOD35_L2/
MOD35_L2.A2003331.0610.004.2003331144742.hdf
```

Figure 6-5. Sample CLEAN work order for the Sweep Data station. Note that individual lines have been split for display purposes here.

```
FileId=/usr/ecs/TS2/CUSTOM/g0spg10/s4ins/terra/reprocessing/DATA/INPUT/
AM1EPHN0.A2000270.0600.001.2001172213441 Uses=9
FileId=/usr/ecs/TS2/CUSTOM/g0spg10/s4ins/terra/reprocessing/DATA/INPUT/
GDAS_0ZF.A2000270.0300.001.2000271023141 Uses=72
FileId=/usr/ecs/TS2/CUSTOM/g0spg10/s4ins/terra/reprocessing/DATA/INPUT/
MOD000.A2000270.0600.001.2000285124229 Uses=9
FileId=/usr/ecs/TS2/CUSTOM/g0spg10/s4ins/terra/reprocessing/DATA/INPUT/
AM1ATTN0.A2000270.0600.001.2001172213440 Uses=9
```

Figure 6-6. Sample INSERT work order for the Track Data station. Note that individual lines have been split for display purposes here.

Stations whose input work orders are in Pathname List format are:

- Sweep Data
- Track Data

6.4.2.5 E-Mail Format

This is not really a format per se, but instead represents different formats that are distributed via e-mail (although other options exist such as ftp). Distribution Notifications and Insert Notifications are sent through e-mail and processed in S4PM via procmail whereas PANs (and their ilk) are ftp pushed.

Stations whose input work orders are in E-Mail format are:

- Receive DN

- Receive PAN
- Subscription Notify

6.5 Data Files

Since in most processing scenarios, S4PM's purpose will be to generate data, it is worth while discussing how data files in S4PM are named and managed.

6.5.1 Data File Names

Prior to release 5.7.0, data files generated in S4PM adhered to a particular file naming convention. In release 5.7.0, a new feature has been introduced that allows other file naming conventions to be used. These other possible file naming conventions will not be discussed here.

The standard S4PM file names follow this convention:

DataType[:RegionID].AYYYYDDD.HHMM.VVV.YYYYDDDHHMMSS[.hdf].

where:

- DataType is the data type name (ESDT ShortName if in ECS)
- [RegionID] is a regional spatial subset identifier. This portion of the name is only included when spatial subsetting is invoked (see Section B.1.9.3.4).
- AYYYYDDD.HHMM is the start date and time of the data file. The 'A' in front is historical baggage and can be ignored. The YYYY is the 4-digit year, the DDD is the day of year, the HH specifies the hour and the MM the minutes.
- VVV is a three-digit data type version (VersionID if in ECS)
- YYYYDDDHHMMSS is the production date/time (in S4PM) in a similar format to the data start date/time, but this time it includes the seconds field.
- [.hdf] is the file name extension assumed for all files generated within S4PM. For files that arrive in S4PM via the Register Data (see Section 18), the file is renamed from its native file name to one following the S4PM file naming convention. The .hdf extension is only applied if the file is really an HDF file; if not, no file name extension is applied.

Some example file names for data generated within S4PM are shown below:

- `AIRIBCBS.A2004244.2359.002.2004246223642.hdf`
- `MOD01.A2004262.1450.004.2004272121209.hdf`
- `MOD021KM.A2004296.0030.004.2004296093719.hdf`
- `AM1ATTH0.A2004294.1200.001.2004294185906.hdf`
- `MOD021SC.A2003244.1655.004.DCB.2004104195431.hdf`

The above file naming convention is used for all output produced by algorithms regardless of whether the data file is really HDF or not.

Using the information above, let's parse the first data file name:

```
AIRIBCBS.A2004244.2359.002.2004246223642.hdf
```

From the above file name, we can deduce the following information:

1. The data type is AIRIBCBS.
2. The data start date is August 31, 2004 (day of year 244 is Aug 31).
3. The date start time is 23:59.
4. The version of data type AIRIBCBS is 002.
5. The data was produced in S4PM on September 2, 2004 (two days after the data for it was collected) at 22:36:42.

The last of the example file names above:

```
MOD021SC.A2003244.1655.004.DCB.2004104195431.hdf
```

is a case where the regional spatial subset identifier is included (the 'DCB').

Some example file names for data imported into S4PM are shown below:

- `PMCOGBAD.A2004295.2200.002.0000000000000`
- `GDAS_0ZF.A2003224.0900.001.2004253203458`
- `NISE.A2004284.0000.001.2004284123102`

The advantage of a consistent file naming convention is twofold: First is transparency. By looking at any file in S4PM, you can tell the data type and version, its temporal coverage, and when S4PM produced it. This makes problems easier to find and diagnose. Second is that S4PM doesn't need additional code to enable it to parse multiple file naming schemes and this keeps S4PM simple (the first S in S4PM!).

It is important to realize that the file names as they exist in S4PM are only seen by S4PM. All algorithms running in S4PM are required to set the LocalGranuleID metadata attribute to the file name the file will have when outside of S4PM. When this is done, file

> **Note:** Even with the new ability in S4PM 5.7.0 to define an alternate file naming convention, the above advantages still hold.

names within S4PM are decoupled from file names that users of the data will see.

6.5.2 Metadata Files

All data files in S4PM are accompanied by metadata files. This is true for files generated within S4PM and those arriving in S4PM from external sources. Two types of metadata files are supported in S4PM: ODL and XML. The ODL (Object Data Language) type is based upon the ECS B0 data model. ODL metadata files have the same name as the data file, but with the .met file name extension added to the end. The XML (eXtensible Markup Language) type also has the same name as the data file, but with the .xml file name extension added to the end. These are examples of data and metadata file pairs:

- NISE.A2004253.0000.001.2004253123101
- NISE.A2004253.0000.001.2004253123101.met

- MOD021KM.A2004296.0010.004.2004296093649.hdf
- MOD021KM.A2004296.0010.004.2004296093649.hdf.met

- AIR2BHA.A2002156.2341.002.2002160234322.hdf
- AIR2BHA.A2002156.2341.002.2002160234322.hdf.xml

All data files that come into S4PM from the outside (*i.e.* via the Register Data station) are assumed to have metadata files associated with them. For data coming from the ECS archive or ECS Datapool, this assumption is met. For data coming from other systems, the providing system must make available at least a minimal metadata file in either ODL or XML format for each data file.

Data generated within S4PM by algorithms must also generate accompanying metadata files. For algorithms using the ECS Toolkit, algorithms are provided an API for generating ODL metadata files. For algorithms not using the ECS Toolkit, S4PM supports tools and methods for wrapping the algorithm by a script that can handle the metadata file generation.

6.5.3 UR Files

A second file associated with all data files is the UR (for Universal Reference, an ECS term) file. UR files are small, one line files with the same name as the data file, but with the .ur file name extension added.

UR files contain a single line consisting of the ECS UR (for files brought into S4PM from the ECS) or the LocalGranuleID for files not coming from ECS and files generated within S4PM.

An example of the contents of a UR file for data coming from ECS is (broken into 2 lines only to fit on the page here):

```
UR:10:DsShESDTUR:UR:15:DsShSciServerUR:13:[GSF:DSSDSRV]
:21:SC:AM1ATTN0.001:82476
```

An example of the contents of a UR file for data generated within S4PM is:

```
LGID:MOD01:004:MOD01.A2000270.0720.004.2004296144737.hd
f
```

The main purpose of UR files in S4PM is to function as a signal file indicating that the data files with which they are associated have been written to completion. A UR file is only created after the data file and metadata file have been written. The utility of UR files is particularly important for large data files. S4PM code that tries to access a data file will not "see" it until a UR file is present. Thus, even if both the data file and metadata file are visible on disk, the Find Data station (Section 12) will not acknowledge its existence until there is also a UR file present.

UR files are generated by S4PM internally and not by algorithms.

Below are examples of data files with their companion metadata and UR files:

```
MOD021KM.A2004296.0005.004.2004296093649.hdf
MOD021KM.A2004296.0005.004.2004296093649.hdf.met
MOD021KM.A2004296.0005.004.2004296093649.hdf.ur

AM1EPHN0.A2000270.0600.001.2001172213441
AM1EPHN0.A2000270.0600.001.2001172213441.met
AM1EPHN0.A2000270.0600.001.2001172213441.ur
```

7. Working with Algorithms

Note: A complete discussion of algorithms and algorithm configuration is contained in S4PM 5.8.1 Installation and Configuration Guide.

This section will describe the basics of working with algorithms in S4PM.

7.1 What Algorithms Can S4PM Support?

Essentially any algorithm code can be supported by S4PM. The following, however, are some things to consider:

1. Algorithms should not assume a particular directory structure. This means that the output file locations, input file locations, and the location from which the algorithm is running should not be hard coded into the algorithm. An algorithm that does hard code these items can be made to work in S4PM, but it requires extra work.

2. Algorithms should produce metadata files for the products they produce. The metadata format is ODL or XML using the EOSDIS data model. Algorithms that don't produce metadata will need to be wrapped by a script that carries out this function for them.

3. An algorithm that requires command line arguments can be handled easily so long as the arguments are static, that is, they don't change from one run to another. If this is not the case, a wrapper script would need to be written that finds and sets the runtime value of any dynamic arguments.

7.2 Algorithm Production Rules

S4PM supports a fairly rich set of production rules that control the inputs that each algorithm sees at runtime. A summary of the production rules supported in S4PM is:

- Basic production of one or more products having the same temporal coverage as the input.
- Time-shifted inputs forward or backward in time relative to the triggering input.
- Time-shifted processing period relative to the triggering input.

- Designation of both required and optional input.
- Multiple alternate inputs, both required and optional, with order or preference specified.
- Wait timers on all inputs (except the triggering input).
- Spatial subsetting whereby all outputs have the same temporal coverages, but are spatially distinct.
- Input accumulation to support daily or multi-day compositing or aggregating algorithms.
- Proxy data types that represent more than one input data type.

7.3 Production Rule Concepts

7.3.1 Simple Production Scenarios

```
INPUT A ⇒ Algorithm ⇒ OUTPUT C
```

The simplest production rule is an algorithm that reads in one data file of data type A and outputs one data file of data type C. Such an algorithm will run every time a data file of data type A arrives. If three such data files arrive at once, three separate runs of the algorithm will be kicked off in S4PM. Here, we assume that the time coverage of the output is the same as the time coverage of the input. Further, we assume each run is completely uncorrelated. Hence, if the input file of data type A has a coverage from Oct 23, 2004 10:00:00 to Oct 23, 2004 10:05:00, the time coverage of the output C will be the same. The above is a description of the most simple production rule.

```
INPUT A
                ⇒ Algorithm ⇒ OUTPUT C
INPUT B
```

A slightly more complex (and realistic) production rule is one that has more than one input. For example, data types A and B are both needed to produce one output file of data type C. In this case, a run of the algorithm will not occur until both data types A and B arrive. We can just as easily have three or more inputs. Likewise, the number of outputs is unrestricted. In fact, an algorithm may produce no output at all (for example, an algorithm that updates a database with a new table row without producing any output file).

```
          INPUT A (Trigger)
```

```
                                        ⇒ Algorithm ⇒
OUTPUT C
INPUT B (1 Step Earlier)
```

In the above examples, we assumed that the time coverage of the output files matched that of the input. But this is not a requirement. In fact, S4PM can support the notion of time-shifted inputs. An algorithm may need one or more of its inputs shifted in time (backward or forward) relative to a data type designated as the trigger data type. For example, if we designate input A as the trigger, an algorithm may require that input B not have the same time coverage as A, but be the one earlier in time.

S4PM further supports optional inputs. An algorithm will not be run unless all of the required inputs are available. If an input data type is designated as optional, the algorithm will look for that data type, but if it cannot be found within a configurable time limit, the algorithm will run without. There can be more than one optional input and these optional inputs can be ordered as to what is the most desired through what is the least desired. S4PM will attempt to use the most desired optional input. If not available, it will attempt to look for the next most-desired input, etc. If none of the optional inputs are available, the algorithm will run without it.

You will often see the term PGE. This simply refers to the algorithm and in this context, PGE is synonymous with algorithm. (PGE actually stood for Product Generation Executive).

7.3.2 The Stringmaker Algorithm Configuration File

The production rules illustrated above and many others are embodied in the Select Data configuration file. Once Select Data configuration is needed or each algorithm. In fact, the Select Data configuration file is part of the algorithm package (discussed below). Here, we discuss this important configuration file and how to generate it.

7.4 Algorithm Installation and Configuration

For details in algorithm installation and configuration, please refer to Section 10 of the S4PM 5.8.1 Installation and Configuration Guide.

8. The Allocate Disk Station

8.1 Station Overview

Is this a component …	Of this S4PM flavor?
✓	Standard Real-Time Processing
✓	Standard Retrospective Processing
✓	On-Demand Processing
✓	Data Mining
✓	With Datapool Polling
✓	With Insert to Datapool

The Allocate Disk station is responsible for allocating disk space from one or more disk pools for a particular run of an algorithm or service. This station is used in all S4PM configurations. The allocated disk pools are then assigned as directories into which the algorithm or service will write its output data. The station allocates disk for the output data only. The space and directory assignments for input data are done when the data are ordered by S4PM in the Register Data (Section 18) station.

S4PM is configured during installation with a set of disk pools which, in aggregation, represent an area of physical disk reserved for use by S4PM. All data moving in and out of S4PM go through these disk pools. The number of disk pools and their sizes are set up during S4PM configuration. The mapping of data type to disk pool is also made during configuration.

A configuration file, s4pm_allocate_disk.cfg, in the Allocate Disk station contains information on the disk pools. It includes the maximum sizes of all data types, the disk pool names to which they are assigned, and the sizes of those disk pools. Based on the data types in the input work order, the Allocate Disk station will attempt to allocate disk from the appropriate disk pool and assign that disk pool as the directory into which the data will be written by the algorithm. If the attempt fails due to no space being currently available, Allocate Disk will simply recycle the work order under the theory that space should become available shortly.

Jobs the Allocate Disk station rarely show up as green or running. Typically, you will only see blue boxes indicating unprocessed work orders. The reason for this is that jobs whose requests for disk space can be satisfied, run very quickly. Conversely, jobs whose requests are denied (because no space is currently available) exit and the work order is recycled immediately.

8.2 Station Work Orders

Standard Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	GETDISK_algorithm	PCF	Prepare Run (Section 14)
	UPDATE_POOLS	N/A	Allocate Disk
Output:	RUN_algorithm	PCF	Run Algorithm (Section 21)
On-Demand Processing			
Input:	GETDISK_service	PCF	Prepare Run (Section 14)
Output:	RUN_service	PCF	Run Algorithm (Section 21)

Table 8-1. Work orders in the Allocate Disk station.

The only distinction in work orders between standard processing and on-demand processing is semantic. In standard processing, the term algorithm is used; in on-demand processing, the convention is to use service instead.

The GETDISK work order is a process control file or PCF. It contains file names and directory names of input data for a single run of the algorithm or service. It also contains output file names, but the output directories are as yet unspecified and instead contain placeholders. Once disk space from a particular disk pool has been allocated, the output directory is set to that disk pool.

The resulting RUN work order is identical to the input GETDISK work order with the exception that the output directories have been fully specified. The output RUN work order is thus a completed PCF.

The UPDATE_POOLS work order is rather unique and was introduced in release 5.6.2. Its purpose is to modify disk pool allocations “on the fly”. This work order is automatically generated whenever Stringmaker is run to reconfigure the string. The work order contents are identical to that of the s4pm_allocate_disk_pools.cfg file.

TIP! If you ever need to increase or decrease any of the disk pools on the fly, simply:

1. Copy the current s4pm_allocate_disk_pools.cfg file to a new file, say: new.cfg

2. Edit new.cfg modifying the disk pools you wish to increase or decrease. Save the file and exit the editor.
3. Verify the syntax of the file by running: `perl -c new.cfg`
4. If the syntax checked out, copy new.cfg to an UPDATE_POOLS work order: `cp new.cfg DO.UPDATE_POOLS.0.wo`
5. The Allocate Disk station should see this new work order, process it and modify the s4pm_allocate_disk_pools.cfg file accordingly. You should verify this. You might also bring up the View Disk Allocation and Usage tool (Figure 8-1) to confirm the changes.
6. Note that if you already have the View Disk Allocation and Usage tool open before you make the changes and your changes include a *reduction* in some disk pool, the tool's display may get confused and show a negative amount. The fix to this is to simply quit and then restart the View Disk Allocation and Usage tool.

8.3 What To Monitor

As mentioned earlier, jobs in the Allocate Disk station run very quickly and job failures should be extremely rare.

To monitor the current state of the disk pools, use the View Disk Allocation and Usage tool which is illustrated below in Figure 8.1.

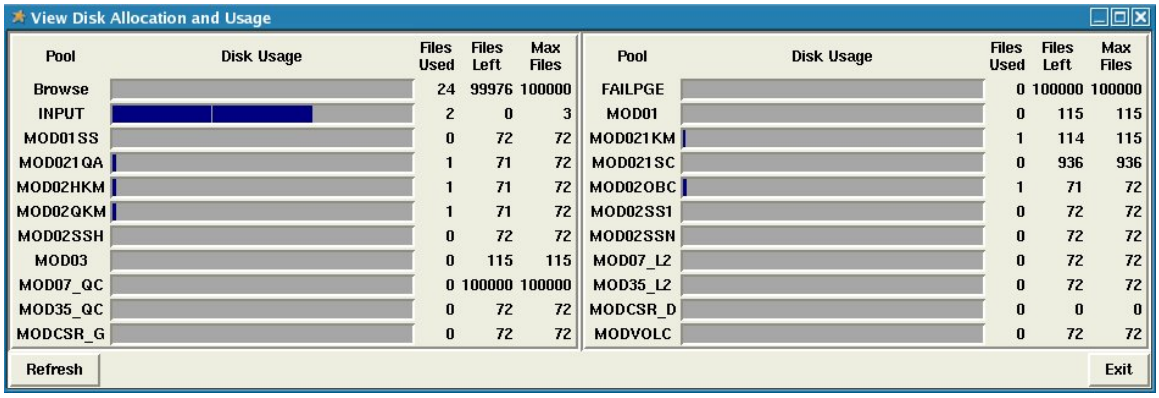

Figure 8-1. The View Disk Allocation and Usage tool window. Disk pool names are displayed down the left side of each pane. Current disk usage is represented by the amount of dark blue. Numbers representing the files used, left, and maximum are shown down the right side of each pane. In this example, the INPUT pool is about two-thirds filled, several other pools have a single file, and the remaining pools are empty. Click on image to zoom in.

Each disk pool is represented in the window with the pool name on the far left. The bar to the right indicates the fraction of the available space in the disk pool that is currently allocated. The three columns on the right side quantitatively list the number of data granules in the pool, the number of additional granules that can be allocated, and the total number of granules that can be allocated. The window is updated periodically, but clicking on the Refresh button will update the window immediately.

8.4 What Can Go Wrong

As mentioned above, jobs that cannot get disk space allocated are immediately recycled. Typically, space will be available the second or third time through. If a disk pool is full, however, the number of times a work order is cycled can grow large. Usually, disk pools filled to capacity is a transient even and should raise no concern.

Sometimes, due to misconfiguration, a job may try to request space in a disk pool that doesn't exist. In this case, the requested space is *never* available and the work order is recycled indefinitely. To check for such occurrences if you suspect them, examine the work order log files.

9. The Auto Request Station

9.1 Station Overview

Is this a component …	Of this S4PM flavor?
☐	Standard Real-Time Processing
☑	Standard Retrospective Processing
☐	On-Demand Processing
☑	Data Mining
☐	With Datapool Polling
☑	With Insert to Datapool

The Auto Request station is only used to provide an automated mechanism for feeding data requests into the Request Data station. This feature is only invoked when the string is configured with the parameter $has_auto_request set to non-zero in the Stringmaker string configuration file.

Although the Auto Request station can be configured in any type of S4PM string, its main purpose is to support retrospective processing strings where data are ordered via the Compose Data Request tool. This is normally a manual operation. The Auto Request station and the accompanying Auto Request Tool automate this process by feeding data requests to the Request Station directly at a rate necessary to achieve some processing rate. The Auto Request Tool window is shown in Figure 9-1 below:

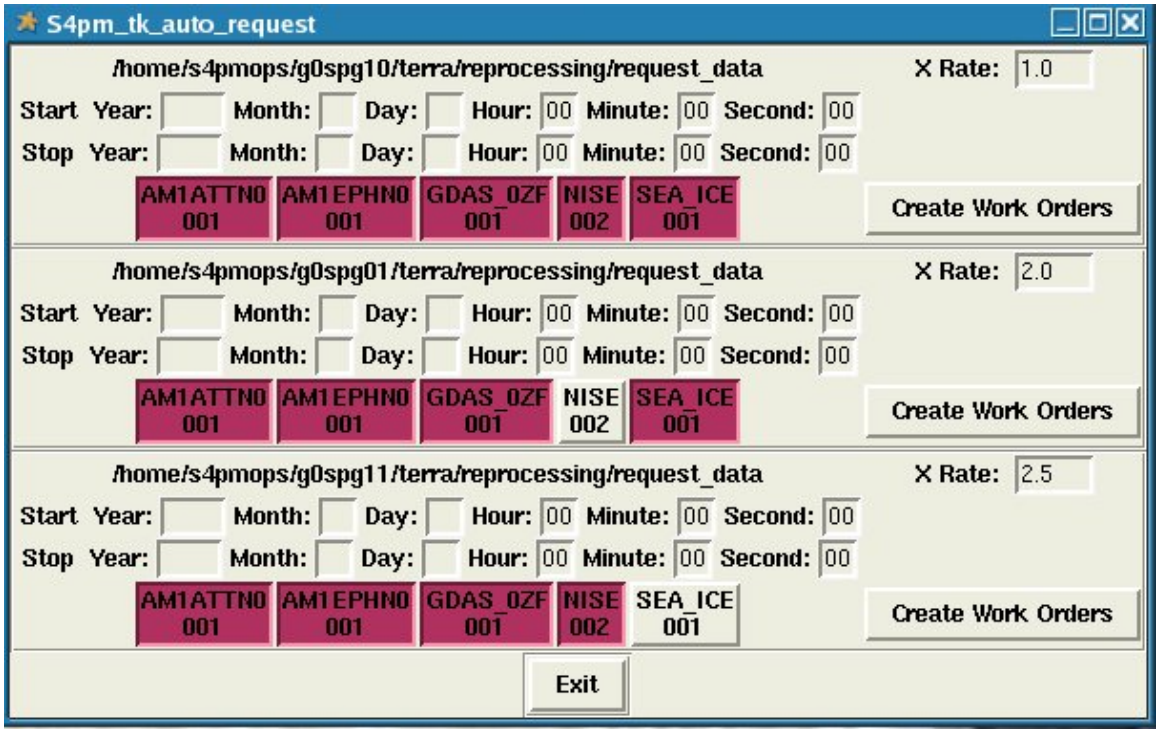

Figure 9-1. An example showing the Auto Request Tool which allows the automated ordering of data via the Request Data stations of multiple strings on a single box.

The Auto Request Tool displays a panel for each string on a particular machine configured for auto request functionality. The Request Data station directory is shown at the top of each panel.

To use, the operator enters in the start and stop dates of data to be ordered for each string desired and then the desired X rate. The **X Rate** is defined so that an X rate of 1.0 means that one hour's worth of data is ordered once per hour; an X rate of 2.0 means that two hour's worth of data are ordered once per hour; etc. These X rates may be configured by an operator to optimize performance on a particular machine (this assumes that all strings running on the machine are configured for Auto Request). Also, note that the X rate controls the rate at which work orders are submitted to the Request Data stations, not the performance rate of the individual strings.

The data types involved in the ordering process are also shown in each panel for each string. These data types may be de-selected by clicking on them. This results in those data types being dropped from any orders (for example, the NISE and SEA_ICE data types in Figure 9-1).

The command for starting up the Auto Request Tool is:

s4pm_tk_auto_request.pl [request_data_station_path]
[request-data_station_path] …

where request_data_station_path is the full or relative path to the Request Data station directory of the string(s) to include in the Auto Request Tool. For example, the command corresponding to the window in Figure 9-1 is:

```
s4pm_tk_auto_request.pl ~/g0spg10/terra/reprocessing/request_data
~/g0spg11/terra/reprocessing/request_data
~/g0spg01/terra/reprocessing/request_data
```

If you include a path to a string that was not configured for Auto Request, the panel for that string will be grayed out and non-functional.

By default, all data types configured as external data types for a string will be displayed in the window. To be more specific about which data types are available, include the –d option on the command line as in:

```
s4pm_tk_auto_request.pl -d AM1ATTNF.001,AM1EPHN0.001
~/g0spg10/terra/reprocessing/request_data
~/g0spg11/terra/reprocessing/request_data
~/g0spg01/terra/reprocessing/request_data
```

See the s4pm_tk_auto_request.pl man page for other options.

9.2 Station Work Orders

Standard Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	AUTO_REQUEST	PCF	Auto Request Tool
Output:	REQUEST_DATA	PDR	Request Data (Section 20)

Table 9-1. Work orders in the Auto Request station.

The AUTO_REQUEST work order is generated by the Auto Request Tool and dropped into the Auto Request station. Based upon the selected X rates and knowledge about the data for the string, one or more REQUEST_DATA work orders are sent to the Request Data station. These REQUEST_DATA work orders are identical to those generated via the Compose Data Request Tool.

10. The Configurator Station

10.1 Station Overview

Is this a component …	Of this S4PM flavor?
✓	Standard Real-Time Processing
✓	Standard Retrospective Processing
✓	On-Demand Processing
✓	Data Mining
✓	With Datapool Polling
✓	With Insert to Datapool

The role of the Configurator station is to provide a string self-configuring functionality. Its main role in release 5.8.1 is to provide a mechanism for installing or uninstalling algorithms on the fly and for modifying the maximum number of jobs a station can run. It is important to realize that there is only one Configurator station across all S4PM strings on all boxes. In this sense, it is similar to the Subscription Notify station (Section 26) except that Subscription Notify is not necessarily a part of all S4PM strings while the Configuration station is.

10.1.1 Installing an Algorithm

To install an algorithm, right-click on the Configurator station button and select **Install Algorithm**. A pop-up window will ask you to select which string to apply the install. An example is shown in Figure 10-1 below:

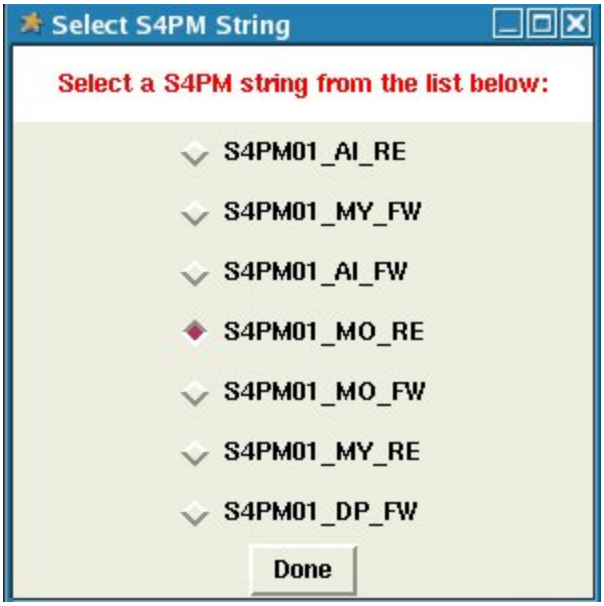

Figure 10-1. An example of the pop-up window you get when you select Install Algorithm or Uninstall Algorithm in the Configuration station.

After making a selection, a new window will ask you to select the algorithm configuration file corresponding to the algorithm you wish to install. See Figure 10.2.

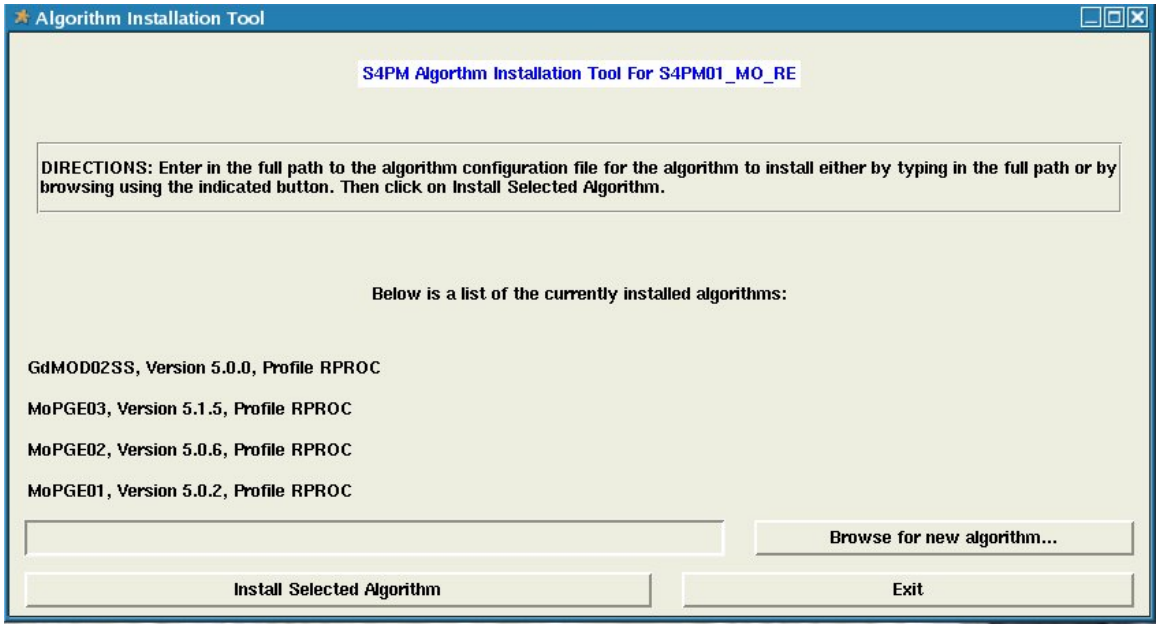

Figure 10-2. An example of the Algorithm Installation Tool window. Algorithms already installed in the string are listed.

To select an algorithm configuration file, you may enter the full path into the entry field provided in the window or select the **Browse for new algorithm…** button to navigate to

the configuration file. Note that you need to select the algorithm configuration file, not merely the algorithm and version directory.

Once selected, click on **Install Selected Algorithm**. In the background, Stringmaker will be run to configure the string for the new algorithm. Afterward, RECONFIG work orders will be dropped into stations whose configurations need to adapt to the new algorithm. These RECONFIG work orders tell the affected stations to re-read their changed configuration files so that the changes take effect. In addition, an UPDATE_POOLS work order will be sent to the Allocate Disk station so that changes in disk pool sizes due to the new algorithm will take effect.

10.1.2 Uninstalling an Algorithm

To install an algorithm, right-click on the Configurator station button and select **Uninstall Algorithm**. The same pop-up window that asks you to select which string to apply the install will now ask you to which string to apply the uninstall. See Figure 10-1.

After making a selection, a new window will ask you to select the algorithm you wish to uninstall. See Figure 10.3.

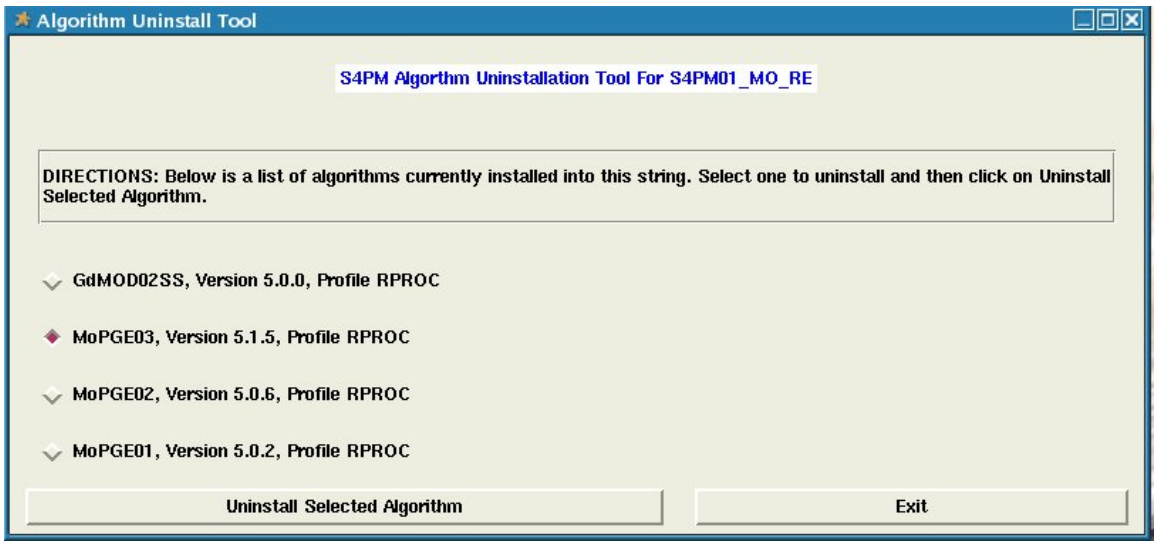

Figure 10-3. Example of Algorithm Uninstall Tool. Algorithms already installed are listed.

Simply select an algorithm and click on the **Uninstall Selected Algorithm** button. As with the installation of an algorithm, Stringmaker will be run behind the scenes to reconfigure the string to run without the selected algorithm. Also as with the installation process, the uninstall process involves an UPDATE_POOLS work order and multiple RECONFIG work orders getting sent to pertinent stations.

Be aware, if you uninstall an algorithm while there are still work orders for that algorithm extent within the system, failures will result since S4PM will have lost all memory of the algorithm uninstalled and hence, any related work orders. These failures should not cause any problems and can be safely deleted.

10.1.3 Modifying Max Jobs

The other task carried out by the Configurator station is to handle changes to the maximum number of jobs run in a station. For most stations, the default is five. For some stations that need to be single-threaded, the default is one.

To modify the maximum number of jobs for a station, right-click on the Configuration station button and select **Modify Max Jobs**. The Modify Max Jobs window will come up and look as shown in Figure 10-4.

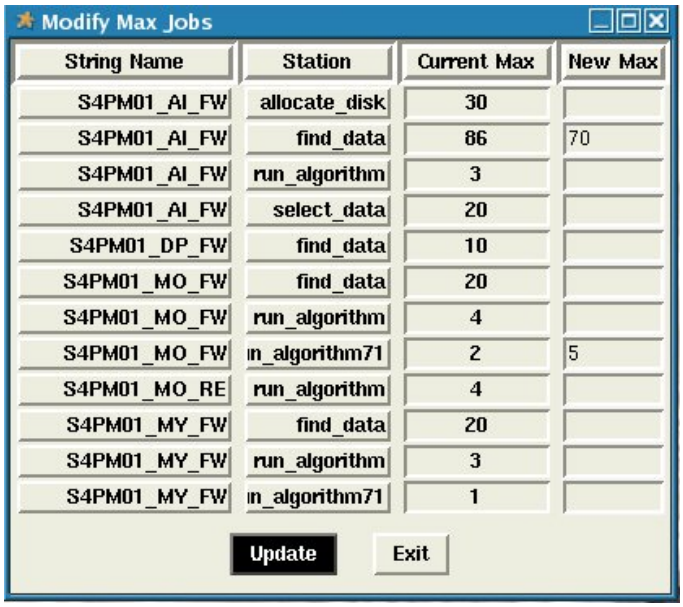

Figure 10-4. An example of the Modify Max Jobs tool. Current values are shown in the next to the last column. Any changes to these values are entered in the last column.

The Modify Max Jobs window will show stations for all strings on the same machine. This allows a more global approach to adjusting maximum jobs since it allows one to maintain a balance across all strings.

Current maximum job settings are shown under the **Current Max** column. To make changes (increasing or decreasing), simply enter new values under the **New Max** column. Then click on the **Update** button to have the changes take effect. Again, Stringmaker will be executed and RECONFIG work orders will be sent to the stations in the strings affected by the changes. There is no need to “bounce” stations to have the changes take effect.

Note that not all stations are listed in this tool. Only those stations that have been configured in the Stringmaker Jobs configuration file are listed. In order to have the ability to modify other stations (via this tool), they first need to be added to this configuration file.

10.2 Station Work Orders

Standard Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	MODIFY_MAX_JOBS	Unique	Configurator
Output:	RECONFIG	N/A	Multiple Stations

Table 10-1. Work orders in the Configurator station.

10.3 What To Monitor

This station is not active most of the time. In fact, it can be kept down until there is a need to install/uninstall an algorithm or modify the maximum number of jobs in a station.

10.4 What Can Go Wrong?

As alluded to earlier, when uninstalling an algorithm, it may be wise to first allow any associated work orders to complete processing. In this way, you will avoid failures when the string is configured to “forget” the algorithm. This station is not active most of the time. In fact, it can be kept down until there is a need to install/uninstall an algorithm or modify the maximum number of jobs in a station.

11. The Export Station

11.1 Station Overview

Is this a component …	Of this S4PM flavor?
✓	Standard Real-Time Processing
✓	Standard Retrospective Processing
✓	On-Demand Processing
✓	Data Mining
✓	With Datapool Polling
✓	With Insert to Datapool

The Export station is responsible for exporting data produced by an algorithm to either the ECS archive or to the ECS Datapool. When configured to export to the ECS archive, the standard SIPS interface is used. The s4pm_export.cfg configuration file sets which data types go to the ECS archive, which go to Datapool, which get deleted (go into the "bit bucket"), and which get distributed to a user (via the External Product Dispatcher mechanism [EPD]; only in on-demand). The input work order is a PDR containing the files to be exported. If the string is configured to send outputs to more than one destination (*e.g.* some outputs to the ECS archive and some to the ECS Datapool), Export will produce multiple output PDRs. Otherwise, it will only produce one PDR. Export places the output PDRs for data going into the archive directly in the ECS PDR polling directory, PDRs for data going to the Datapool directly in the Insert Datapool (Section 13) station directory, and PDRs for EPD into a directory where ECS is polling for EPD PDRs. Since Export directly places output PDRs into the proper directories, there is no output work order per se from this station (only if Stationmaster moves the work order is it considered a true work order). Log files, however, are configured to go into the ARCHIVE directory under the station root directory.

Upon ingest into the ECS archive, insert into the ECS Datapool, distribution to the user, or deletion, Product Acceptance Notifications or PANs are sent to the Receive PAN (Section 17) station. PANs for data that are deleted are actually generated by the Export station itself to mimic PANs generated by ECS.

There is no Export station in on-demand processing. Rather, data are distributed directly to requesters via the Distribution Command Line Interface or DCLI.

For testing purposes, the Export station can be configured for "fake" export to the ECS archive. In this set up, the response of the ECS is mimicked at a crude level; the data exported are simply deleted.

11.2 Station Work Orders

Standard Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	EXPORT, EXPORT_BROWSE, EXPORT_PH, EXPORT_EPD, EXPORT_FAILPGE	PDR	Run Algorithm (Section 21)
Output:	None	PDR	ECS PDR Polling
	None	PDR	Insert Datapool (Section 13)

Table 11-1. Work orders in the Export station.

The input EXPORT work orders are in PDR format and are sent to the Export station by the Run Algorithm (Section 21) station following the successful run of an algorithm.

There is no upstream station from Export and hence, there is no output work order produced. In standard S4PM configurations, however, the EXPORT log file is typically sent to the ARCHIVE/logs/export directory. This log is a chain log file containing the entire thread of station activities leading to the products being exported.

11.3 What To Monitor

The Export station doesn't have a lot of work to do beyond some minimal checking of the input PDR work orders and moving them to where ECS ingest or ECS Datapool insert can see them. Therefore, jobs should run very quickly in this station and should not linger more than several seconds.

11.4 What Can Go Wrong

Only very rarely does the Export station ever fail.

12. The Find Data Station

12.1 Station Overview

Is this a component …	Of this S4PM flavor?
✔	Standard Real-Time Processing
✔	Standard Retrospective Processing
✔	On-Demand Processing
✔	Data Mining
✔	With Datapool Polling
✔	With Insert to Datapool

The Find Data station is responsible for locating the data needed to run an algorithm. The Find Data station will poll the S4PM data directories periodically. Polling of the disk pools occurs every 30 seconds by default. If “smart” polling has been enabled, the polling frequency can actually be configured to change over time (typically, polling becomes less frequent the longer a job has been running). If smart polling has been configured, you will see the –smart command line option on the command line where s4pm_find_data.pl is referenced in the station.cfg file. Regardless of the polling frequency, between pollings the job sleeps.

The Find Data station is a component of all S4PM configurations.

The input work order is a PDR that contains the data files being requested by the algorithm. Data files in the work order are indicated as being either required or optional. If required, the Find Data station job will fail if the data cannot be found within a configurable time limit. If optional, the Find Data station job will nonetheless succeed.

The Find Data station actually handles far more complex rules than indicated above. These complex rules are usually referred to as production rules. Production rules allow wait timers to be associated with each requested input, beyond which the Find Data station gives up on looking for the data. In addition, degrees of optionality can be described so that one type of data is the most desired option, another is next, another is the even less desired, and so on.

The wait timers mentioned above work this way: First, the timer starts at zero once the job in Find Data station begins. This is essentially the time that the trigger data type for the algorithm arrives. Once all of the required input data have been located, the timer is reset to zero again. Find Data then begins looking for the optional input data (if any).

In standard processing (real-time or retrospective), the Find Data station attempts to find files using a file name pattern. This is because the precise file name is not known and is not specified in the input PDR work order. What is known about the requested file is the data type, data type version, start date, and start time. With these elements, Find Data can form a file name pattern with which to search for the data file. This is because all files in standard processing configurations adhere to a single file name convention. If you examine a PDR work order, you will see INSERT_FILE_HERE for the FILE_ID attribute and INSERT_DIRECTORY_HERE for the DIRECTORY_ID attribute. These are placeholders. Once Find Data has completed successfully, it will replace these placeholders with actual file names and directories.

In on-demand processing, the situation is different. The input PDR work order contains the actual file names and directories, not placeholders. Thus, Find Data does not have to work from a file name pattern. In addition, there is no concept of optional input data in on-demand processing (at least, not yet). This greatly simplifies the processing.

Refer to the S4PM 5.8.1 Installation and Configuration Guide for more details on algorithm configuration.

12.1.1 Manual Overrides

The Find Data station provides several manual overrides for controlling the jobs. A **Ignore Required** button is provided to force the job to disregard any required data that it hasn't already found. This may be useful in rare cases where it is known that the remaining required input file will never arrive and the algorithm can still produce something useful (despite the fact that it was marked as required).

The **Ignore Optional** button accomplishes the same thing for optional input data. This is more useful since the data are designated as optional indicating that the algorithm can run successfully without it.

The **Expire Current Timer** button simply forces the timer on the data currently being sought to jump to the maximum allowed for that data (this is set in the input PDR work order). The effect is to cause the job to move on to looking for the next item. This can be useful, again, if you know that data will never arrive and you don't want the timer to expire naturally (say, if it is particularly long).

Note: As of release 5.8.1, the Find Data station in on-demand processing is configured (by default) to recycle work orders if data aren't found rather than continually polling. Thus, manual overrides generally do not work.

12.2 Station Work Orders

Standard Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	MOREDATA_algorithm	PDR	Select Data (Section 22)
Output:	TRIGGER_algorithm	PDR	Prepare Run (Section 14)
On-Demand Processing			
Input:	MOREDATA_service	PDR	Split Services (Section 24)
Output:	TRIGGER_service	PDR	Prepare Run (Section 14)

Table 12-1. Work orders in the Find Data station.

The only distinction in work orders between standard processing and on-demand processing is semantic. In standard processing, the term algorithm is used; in on-demand processing, the convention is to use service instead.

The MOREDATA work order is a PDR. In standard processing, it is sent by the Select Data (Section 22) station; in on-demand processing, it is sent by Split Services (Section 24). In standard processing, the PDR contains placeholders for the file names and directories; in on-demand, these items are fully qualified.

The output work order is a TRIGGER work order that is sent to the Prepare Run (Section 14) station. In standard processing, this output TRIGGER_algorithm work order is the same as the input MOREDATA_algorithm work order, except that the placeholders have been replaced by actual file names and directories. In on-demand processing, the output TRIGGER_service work order is essentially identical to the input MOREDATA_service work order.

12.3 What To Monitor

Jobs in Find Data can run for a long time, probably more so than in any other station. The station can be configured to have a job box turn from green to yellow if it has been running beyond a configurable set time. If you see a job turn yellow, it might mean that something is taking too long (e.g. a particular input is taking an unusually long time to arrive). It might also mean that the timer for turning yellow is unrealistically low. If you see yellow boxes often for a particular algorithm, such is the case.

As in indicated earlier, if all required data has not been found, the job fails and the job box turns red (unless you activated the **Ignore Required** button mentioned above). If

after the failure, the missing required data arrives, the job can be restarted and it should complete successfully fairly quickly.

12.3.1 Sleep Message File

The sleep message is a small file in the job directory. It has the file name:

`sleep.message`.process_id

In this file, the current file being sought and the timer for seeking it are listed. An example for standard processing S4PM strings is shown in Figure 12-1 below:

```
Sleeping 30 secs: Awaiting OPTIONAL (OPT1) granule of AIRIBRAD from
2002-10-19T22:17:26Z to 2002-10-19T22:23:26Z to arrive.

Current Clock: 62957

Giving up when clock exceeds: 172800
```

Figure 12-1. Sample contents of a sleep message file in the Find Data station. In this example, Find Data is waiting for an optional data file of data type AIRIBRAD with the time coverage as listed. The job will give up on this data after waiting a total of 48 hours (172800 seconds). So far, 62957 seconds have elapsed.

This message file is updated every polling cycle. In the S4PM Job Monitor window, simply click on the file to refresh your view of the file.

In on-demand processing, the sleep message is somewhat different since the actual file name being sought is known. See Figure 12-2 below:

```
Sleeping 30 secs: Awaiting REQUIRED granule with name
AIR20SCI.A2002293.2200.001.2002294054853 to arrive.

Current Clock: 120

Giving up when clock exceeds: 600
```

Figure 12-2. Sample contents of a sleep message file in the Find Data station in On-Demand processing. Here, the actual file name being waited on is known.

12.3.2 Chain Log File

An important place to look if you suspect a problem is the chain log file, also in the running job directory. Its file name will be: MOREDATA_algorithm_name.jobid.log where algorithm_name is the algorithm name and jobid is the job ID which uniquely identifies the running job. An example for the algorithm AiL1B_AIRS would be: MOREDATA_AiL1B_AIRS.2002292233526.log.

Viewing the chain log file in the S4PM Job Monitor window will show the bottom of the file. This file is constantly being updated; click on the file name again to refresh your view of the contents.

The chain log has a lot of information about what Find Data is currently doing. In particular, look for messages like these:

```
Current Clock: 62685

Giving up when clock exceeds: 172800

2004-05-11 10:52:37 1000886 INFO  s4pm_find_data.pl: get_best_data:
Looking for best data among 1 choices for LUN 6211 starting with the
most desirable.

2004-05-11 10:52:37 1000886 INFO  s4pm_find_data.pl: get_best_data:
Choice: 1: Looking for OPTIONAL granule of AIRIBRAD from 2002-10-
19T23:29:26Z to 2002-10-19T23:35:26Z with need of OPT1 and currency of
PREV1 to arrive.

2004-05-11 10:52:37 1000886 ERROR s4pm_find_data.pl: get_data_pathname:
No file found matching file name pattern: [AIRIBRAD.A2002292.2329.002.*]

2004-05-11 10:53:07 1000886 INFO  s4pm_find_data.pl:
fill_in_optional_file_groups: Awaiting OPTIONAL (OPT1) granule of
AIRIBRAD from 2002-10-19T23:29:26Z to 2002-10-19T23:35:26Z to arrive.
```

Figure 12-3. Sample of a chain log file for a job running in the Find Data station. The bottom of the file is shown. Viewing this file may provide help in diagnosing any problems encountered.

In the Figure 12-3 above, the current clock and the clock number at which Find Data will give up looking for the data file are listed. In addition, the actual file name pattern being used in seeking the data file is displayed. This can be useful if you believe that the desired file has arrived yet Find Data doesn't seem to acknowledge it. You can copy the very file name pattern listed here and see if the file does exist using that pattern. This is only a short view of the types of information one can find in this log file.

12.4 What Can Go Wrong

Being a fairly complex station, a number of things can go wrong in Find Data.

12.4.1 Job stays green seemingly forever.

A job in the Find Data station will run until all required data have been located or until it times out. Some algorithms may be configured for very long wait timers (days or weeks). Therefore, the real question is why haven’t the data arrived when expected. Addressing this question is outside the scope of S4PM.

12.4.2 Data file exists, but Find Data cannot see it.

In this case, the sleep message lists the data type and start/stop times for data that does exist, but Find Data doesn't seem to acknowledge it. Here, the problem is usually that the file name pattern with which Find Data is seeking the data is not what you might think. The only way to be sure is to view the chain log file and try out the file name pattern

yourself in the directory where you "see" the data. Most likely, you will find that the data does not exist according to that file name pattern.

The resolution might be to acquire the missing data or to reconfigure the production rules for that algorithm so that the file name pattern Find Data employs matches your expectations.

12.4.3 Job failed and then the required data arrives.

Here, the job timed out not finding all the required data. But afterwards, the data arrived. The solution here is to simply restart the failed job by clicking on the **Restart** button in the S4PM Job Monitor window.

12.4.4 Job somehow succeeded without all the required data.

The only way this can happen is if someone clicked on the **Ignore Required** button in the S4PM Job Monitor window. You can verify whether or not this happened. In one of the upstream stations from Find Data, examine the chain log file corresponding to the job in question. In the Find Data section of that file towards the end of that section you should see something like:

```
2004-05-11  12:13:07  1090186  WARNING  fill_in_required_file_groups:
Detected a IGNORE_REQUIRED signal file in run directory. Job will be
released without this input. I hope you know what you're doing.
```

Here, we clearly see that someone clicked on **Ignore Required**. There is no way to recover from this situation once the job has succeeded in Find Data.

There is an analogous situation that can occur with optional data if someone clicks on the **Ignore Optional** button. Again, refer to the chain log file to verify this.

Note that to add some level of restriction on actions like Ignore Required and Ignore Optional, you may want to configure the string with privileged users so that only privileged users (that is, operators) can execute these functions. Refer to the `@privileged_users` parameter in the S4PM 5.8.1 Installation and Configuration Guide.

13. The Insert Datapool Station

13.1 Station Overview

Is this a component …	Of this S4PM flavor?
☐	Standard Real-Time Processing
☐	Standard Retrospective Processing
☐	On-Demand Processing
☐	Data Mining
☐	With Datapool Polling
☑	With Insert to Datapool

The Insert Datapool station is tasked with inserting data directly to the ECS Datapool. It can equally well insert to any disk resource whose directory structure is similar to that of Datapool.

Products inserted to the Datapool are handled via an interface provided by ECS for non-ECS data. Thus, such products do not have ESDTs associated with them.

Since the Export (Section 11) station places input PDRs directly into the Insert Datapool directory, there is no input work order *per se* for this station.

In addition, there is no output work order from the Insert Datapool station. The result of a successful insert into the Datapool is a PAN which is placed directly in the Receive PAN (Section 17) station directory (stationmaster normally moves work orders). A failure PAN is created if all inserts fail and a long PAN is created if some fail while others succeed.

An important distinction of the Insert Datapool station is that it must be run on the machine attached to the ECS Datapool.

13.2 Station Work Orders

Standard Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	None	PDR	Export (Section 11)
Output:	None	PAN	Receive PAN (Section 17)

Table 13-1. Work orders in the Insert Datapool station.

14. The Prepare Run Station

14.1 Station Overview

Is this a component …	Of this S4PM flavor?
✔	Standard Real-Time Processing
✔	Standard Retrospective Processing
✔	On-Demand Processing
✔	Data Mining
✔	With Datapool Polling
✔	With Insert to Datapool

The Prepare Run station prepares the running of a particular algorithm. It does so by preparing a process control file (PCF) for the algorithm that contains the specific input and output data files mapped to logical unit numbers. For algorithms that use the ECS Toolkit, input and output files are referenced by these logical unit numbers. In this way, the algorithms maintain an abstracted view of the data they read or write. In addition, static parameters called user-defined runtime parameters can be configured in the PCFs.

For algorithms that do not use the ECS Toolkit, a simple wrapper script can be built using S4PM functions to interface directly with the PCF.

In on-demand processing, the Prepare Run station may in addition add to the PCF the specialized criteria values from the original request (if any). They will appear in the PCF as additional user-defined runtime parameters. Some algorithms may be configured to read the specialized criteria values in the original request called a PSPEC file. This ODL formatted file is always referenced in the PCF.

The PCFs built by the Prepare Run station are unique for each run. Their construction, however, is based on an algorithm-specific PCF template file. This template file is part of the algorithm package and resides in the directory where the algorithm is installed.

14.1.1 Modify PCF Runtime Parameters Tool

The Modify PCF Runtime Parameters Tool allows runtime parameters in the process control file (PCF) associated with an algorithm to be modified. Each algorithm may have any number of runtime parameters defined (or none at all). Although most of these parameters are intended to be "fixed" while in operations, there may be several where it makes sense to modify them *on the fly* during operations. This tool should not be used without fully understanding the impact of the proposed change.

Start the Modify PCF Runtime Parameters Tool by right clicking on the Prepare Run station button and selecting the tool. The first window that comes up is illustrated in Figure 12-1 below.

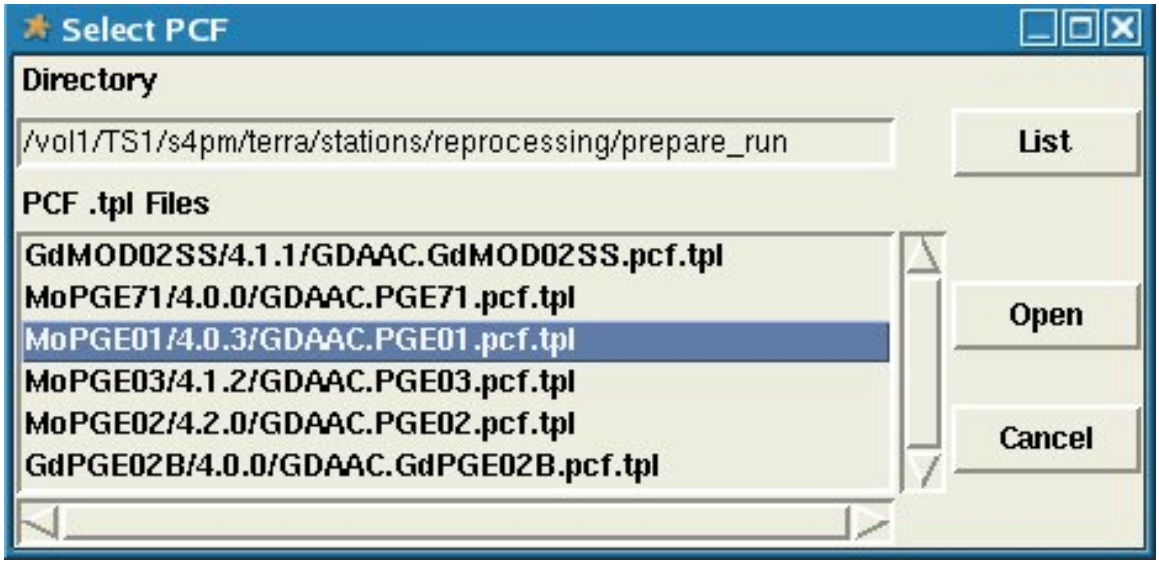

Figure 14-1. Image of the Select PCF window of the Modify PCF Runtime Parameters tool.

In the Select PCF window, the directory containing the PCFs is indicated in the top field. The tool will open in the default location. Should this directory need to be modified, you can modify it in place in then click on the **List** button. The main list box contains the PCFs found in the directory listed above. To modify a PCF, select the desired one from the list and then click on **Open**. An example of the resulting window is shown below in Figure 14-2.

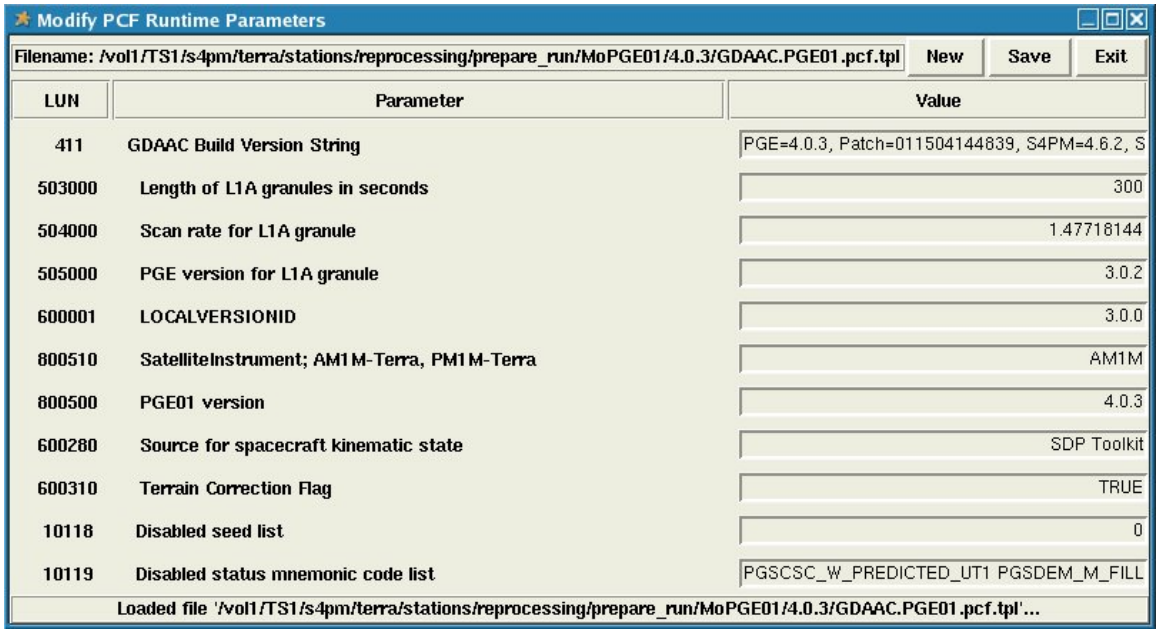

Figure 14-2. Image of the Modify PCF Runtime Parameters window of the Modify PCF Runtime Parameters tool.

The Modify PCF Runtime Parameters window displays the runtime parameters available for modification for the algorithm whose PCF was selected above. Note that this tool doesn't know which parameters should or should not be modified. Therefore, the user needs to be aware of the impact from any changes made. The runtime parameter logical unit numbers (LUNs) and descriptions are listed down the left side and their current settings are listed on the right. To modify any parameter, make the desired changes in the appropriate boxes on the right side. When done, click on **Save**. The changes will then be saved.

Changes made to a runtime parameter only take effect for new jobs going into the Select Data (Section 22) station.

14.2 Station Work Orders

Standard Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	TRIGGER_algorithm	PDR	Find Data (Section 12)
Output:	GETDISK_algorithm	PCF	Allocate Disk (Section 8)
On-Demand Processing			
Input:	TRIGGER_service	PDR	Find Data (Section 12)
Output:	GETDISK_service	PCF	Allocate Disk (Section 8)

Table 14-1. Work orders in the Prepare Run station.

The input work order is a TRIGGER_algorithm or TRIGGER_service. These work orders are PDRs. The input file names and directories are fully specified.

The output GETDISK_algorithm or GETDISK_service work orders are process control files (PCFs). These PCFs contain the input file names and directories (extracted from the input PDR) and output file names which are assigned by the station. The output directories, however, are omitted. Instead, there are placeholders. It will be the job of the Allocate Disk (Section 8) station to replace these placeholders with actual directory names once space for the output has been allocated.

14.3 What To Monitor

Jobs in Prepare Run should run very quickly. Any jobs that linger should be suspect.

14.4 What Can Go Wrong

In building a runtime PCF, the Prepare Run does a check to verify that input data exists. Very rarely, a data file may be deleted between the time that the Find Data (Section 12) station finds it and the time that Prepare Run gets to it. This may happen if the system becomes very backed up and data are deleted after they become several days old. A message in the chain log file like this:

```
main: Missing required input file(s):
/usr/ecs/OPS/CUSTOM/g0spg01/s4ins/aqua_modis/reprocessing/DATA/MYD03/MYD
03.A2003226.1850.004.2004145140052.hdf
```

usually indicates this condition. The only recourse is to reorder the data via the Compose Request Tool. If the problem becomes chronic, it may be worth considering increasing the expiration time for the data type in question.

15. The Poll Data Station

15.1 Station Overview

Is this a component …	Of this S4PM flavor?
☐	Standard Real-Time Processing
☐	Standard Retrospective Processing
☐	On-Demand Processing
☐	Data Mining
☑	With Datapool Polling
☐	With Insert to Datapool

The Poll Data station does what its name implies. It periodically polls a disk area for the arrival of new data and represents another way of getting data into S4PM. Although originally designed for the ECS Datapool, this station can actually poll any disk area whose directory structure is similar.

15.2 Station Work Orders

Standard Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	POLL	N/A	N/A
Output:	TRIGGER_DATA	PDR	Register Data (Section 18)

Table 15-1. Work orders in the Poll Data station.

16. The Receive DN Station

16.1 Station Overview

Is this a component …	Of this S4PM flavor?
☑	Standard Real-Time Processing
☑	Standard Retrospective Processing
☑	On-Demand Processing
☑	Data Mining
☐	With Datapool Polling
☑	With Insert to Datapool

The job of the Receive DN station is to process Distribution Notifications (DNs) from ECS which are sent when data from ECS has been distributed to S4PM. DNs are sent via e-mail and, as with Insert Notifications, are disseminated to the proper S4PM string using procmail.

The DN contains information on what requested data were distributed successfully and what were not due to some failure. If any data fail to get distributed, the entire job fails. When this happens, the **Resubmit Request** button on the S4PM Job Monitor window can be clicked. This will cause a new request to be submitted for only those data that failed the first time to get distributed. The data that succeeded, if any, will pass successfully into S4PM.

The Receive DN station is a component of all S4PM strings except those that poll **all** data from the Datapool.

16.2 Station Work Orders

Standard Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	Distribution Notification	E-Mail	ECS
Output:	TRIGGER_DATA	PDR	Register Data (Section 18)

Table 16-1. Work orders in the Receive DN station.

16.3 What To Monitor

Jobs in Receive DN should run very quickly. Any jobs that linger could indicate problems.

16.4 What Can Go Wrong

Failures in the Receive DN station typically happen when the DN itself indicates a failure of the ECS to distribute the requested data. This is something that must be resolved on the ECS side. Once the ECS problem has been resolved, clicking on the Resubmit Request button in the S4PM Job Monitor window will cause the data request to be resubmitted. Hopefully, the resulting DN will succeed.

17. The Receive PAN Station

17.1 Station Overview

Is this a component …	Of this S4PM flavor?
☑	Standard Real-Time Processing
☑	Standard Retrospective Processing
☑	On-Demand Processing
☑	Data Mining
☐	With Datapool Polling
☑	With Insert to Datapool

The Receive PAN station is responsible for closing the loop on data products exported to the ECS, Datapool, or data to be deleted via the Export station. It does so by waiting for a Product Acceptance Notification (PAN) indicating successful ingestion of data into ECS, Datapool or deletion. After receipt of the PAN, an UPDATE work order is set to the Track Data (Section 28) station notifying it that the data are no longer needed in S4PM.

A short PAN is received by the Receive PAN station when data are successfully ingested, sent to Datapool, or deleted. If the dispositioning of the data is only partially successful (i.e. at least one of the data files failed), a long PAN is received instead. The long PAN includes a list of which files were successfully dispositioned and which were not along with an error mnemonic as to why the ingest failed.

If specifically ECS ingest fails for all data files, a Product Delivery Record Discrepancy (PDRD) may be received by Receive PAN. This usually indicates a syntax bug in the PDR or that ECS ingest is improperly configured for the data types being sent to it.

In the case of a long PAN or a PDRD, the job fails in Receive PAN. If the problem is remedied, the correct action to take is to click on the Resubmit PDR button in the S4PM Job Monitor window. If the failure was a long PAN, only those data that failed to be ingested will be attempted again; the data that were already successfully ingested will not be re-ingested. If the failure was a PDRD, the entire original PDR will be resubmitted for ingest.

PDRs for Production History files and other associated data types (meaning those that are linked to other data types) are held in a PDR_LIMBO directory until the data they are associated with are successfully ingested into ECS. When a successful PAN for these data is received by Receive PAN, these sequestered PDRs are released from PDR_LIMBO to the Export (Section 11) station. Assuming the Production History files or other associated data are successfully ingested, a short PAN for them will be sent to the Receive PAN station.

The PAN mechanism is used also for data going directly into the ECS Datapool, data getting distributed via EPD to users, and even for data being deleted.

Rather than the PANs being sent from the ECS Ingest subsystem, they are instead sent by the Insert Datapool (Section 13) station, by EPD, or by Receive PAN itself in the case of data being deleted.

The Receive PAN station is a component of standard processing only. It is not used in on-demand processing.

17.2 Station Work Orders

Standard Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	PAN, PDRD	E-Mail	ECS Ingest
			Insert Datapool (Section 13)
			External Product Dispatcher (EPD)
Output:	UPDATE	Pathname	Track Data (Section 28)

Table 17-1. Work orders in the Receive PAN station.

In the case of ECS ingest, the input work orders are PANs (long or short) or PDRDs sent by the ECS Ingest subsystem via e-mail. The e-mailed PANs and PDRDs are addressed to the S4PM user who owns and runs the S4PM string. A procmail filter is then used on receipt of these e-mails to move them into the Receive PAN station which is configured to poll for them.

In the case of PANs from Insert Datapool (Section 13), the delivery mechanism is like that of any other work order. It is handled by Station Master as a station to station transfer; e-mail is not used.

The output work order is an UPDATE work order sent to the Track Data (Section 28) station. The format is a simple list of full pathnames of the data that have been successfully ingested. Track Data station can then have these files safely deleted from the S4PM system.

17.3 What To Monitor

The Receive PAN jobs themselves run very quickly. There should not be any lingering green boxes in the station.

18. The Register Data/Register Local Data Stations

18.1 Station Overview

Is this a component …	Of this S4PM flavor?
✓	Standard Real-Time Processing
✓	Standard Retrospective Processing
✓	On-Demand Processing
✓	Data Mining
✓	With Datapool Polling
✓	With Insert to Datapool

The Register Data and Register Local Data stations are responsible for registering data from external sources or those generated within S4PM itself. In standard processing strings, the Register Data station renames the files it registers following a consistent file naming convention. In Register Local Data the original file names are retained.

In on-demand processing, the original file names are retained by both Register Data and Register Local Data stations.

In addition to possibly renaming files it receives, Register Data also creates a UR file. The UR file name is the same as the data file name (after the renaming) with the .ur file name extension added. The UR file serves two purposes. For large files, its creation signals that the file is not actively being written to. This is particularly important with the transfer of large files from external sources. Secondly, the contents of the UR file is a single line containing either the UR as registered in the ECS or the LocalGranuleID from the metadata.

18.2 Station Work Orders

18.2.1 Register Data

Standard Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	TRIGGER_DATA	PDR	Receive DN (Section 16)
Output:	INSERT	Pathname	Track Data (Section 28)
	NEWDATA_algorithm	PDR	Select Data (Section 22)
On-Demand Processing			
Input:	TRIGGER_DATA	PDR	Receive DN (Section 16)
Output:	INSERT	Pathname	Track Data (Section 28)

Table 18-1. Work orders in the Register Data station.

18.2.2 Register Local Data

Standard Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	TRIGGER_DATA	PDR	Run Algorithm (Section 21)
Output:	INSERT	Pathname	Track Data (Section 28)
	NEWDATA_algorithm	PDR	Select Data (Section 22)
On-Demand Processing			
Input:	TRIGGER_DATA	PDR	Run Algorithm (Section 21)
Output:	INSERT	Pathname	Track Data (Section 28)

Table 18-2. Work orders in the Register Local Data station.

19. The Repeat Hourly/Repeat Daily Stations

19.1 Station Overview

Is this a component …	Of this S4PM flavor?
✔	Standard Real-Time Processing
✔	Standard Retrospective Processing
✔	On-Demand Processing
✔	Data Mining
✔	With Datapool Polling
✔	With Insert to Datapool

The Repeat Hourly and Repeat Daily stations provide a mechanism for running pseudo cron jobs in S4PM on an hourly or daily basic, respectively. The types of jobs run in these stations are mostly clean up jobs, for example, jobs that clean up data that aged on disk beyond a configurable limit. Many of the jobs running in these two stations are UNIX find command that look for files matching particular criteria and then doing something with them, usually deleting them.

Each type of activity that the Repeat Hourly and Repeat Daily stations are configured to run are associated with a unique work order type. For instance, the work order REPEAT_CLEAN_FILES is associated with the task of running a script that cleans out data that have aged beyond a configurable amount of time. In this particular case, the job runs in the Repeat Hourly station and, hence, runs once every hour.

After completing a job, the initial input work order is recycled back into the Repeat Hourly or Repeat Daily station so it will be there to trigger the next cycle of that task.

The table below lists the built in tasks that run in the Repeat Hourly and Repeat Daily stations:

Task Description	Work Order Name	Station
Deletion of old data beyond a configurable age.	REPEAT_CLEAN_FILES	Repeat Hourly
Run script that rolls up individual algorithm run statistics into summaries by date.	ROLLUP_RUSAGE	Repeat Hourly
Find and remove Browse product data beyond a configurable age.	CLEAN_BROWSE	Repeat Daily
Find and remove failed PGE (algorithm) tar files beyond a configurable age.	CLEAN_FAILPGE	Repeat Daily
Find and remove production history (PH) tar files beyond a configurable age.	CLEAN_PRODHIST	Repeat Daily
Find and remove data request stub files beyond a configurable age.	CLEAN_REQ_STUBFILES	Repeat Daily
Find and remove files in the ARCHIVE directory beyond a configurable age.	CLEAN_STATION_ARCHIVE	Repeat Daily
Run a script that updates the Show PGE Web tool with the latest information on algorithms running in the S4PM string.	SHOW_PGE	Repeat Daily

Table 19-1. Repeat Hourly and Repeat Daily station tasks built into S4PM.

19.2 Station Work Orders

All Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	REPEAT_CLEAN_FILES, ROLLUP_RUSAGE, CLEAN_BROWSE, CLEAN_FAILPGE, CLEAN_PRODHIST, CLEAN_REQ_STUBFILES, CLEAN_STATION_ARCHIVE, SHOW_PGE	N/A	Repeat Hourly, Repeat Daily
Output:	REPEAT_CLEAN_FILES, ROLLUP_RUSAGE, CLEAN_BROWSE, CLEAN_FAILPGE, CLEAN_PRODHIST, CLEAN_REQ_STUBFILES, CLEAN_STATION_ARCHIVE, SHOW_PGE	N/A	Repeat Hourly, Repeat Daily

Table 17-2. Work Orders in the Repeat Hourly and Repeat Daily stations. Note that since work orders are recycled in these stations, the input and output work orders are the same.

The work order types are shown in Table 17-1 and all are zero length files. It is merely the name of the work order that triggers the hourly or daily task to run. Since all these work orders are recycled, the input and output work orders are formally the same.

19.3 What To Monitor

Most of the time, work orders in these stations will show as blue. The jobs run only hourly or daily and the execution times for these jobs are generally very short.

Note that if, for some reason, these stations have not be running for a long period of time, the clean up jobs may take a very long time when first started up since a very large number of files may have accumulated.

20. The Request Data Station

20.1 Station Overview

Is this a component …	Of this S4PM flavor?
☑	Standard Real-Time Processing
☑	Standard Retrospective Processing
☑	On-Demand Processing
☑	Data Mining
☐	With Datapool Polling
☑	With Insert to Datapool

The Request Data station's role is to request data from the ECS archive. It does so by using the ECS-provided SDSRV (Science Data Server) Command Line Interface tool or SCLI. Using the SCLI, the Request Data submits one or more requests for data that the ECS pushes directly into the input data directory. A Distribution Notification (DN) is then sent by ECS to mark the successful push of the data.

Requests for data originate in one of two places. In standard real-time processing S4PM strings, S4PM subscribes to insert notifications, e-mails that notify S4PM of the insert of new data into the ECS archive. In S4PM, these notifications are intercepted by the Subscription Notify station who constructs a PDR containing the data to request. That PDR is then sent to the Request Data station to make the actual request via the SCLI.

Alternatively, in standard real-time processing or in standard retrospective processing strings, a user can place an order for specific data residing in the ECS archive using the Compose Request tool (see Section 18.1.1). This tool generates the same PDR as would be generated by the Subscription Notify station. The Compose Request tool is shown in Figure 18-1.

In on-demand processing, the job of constructing a PDR of the requested data falls to the Split Services (Section 24) station. Split Services parses the request ODL for the data to request.

20.1.1 Compose Request Tool

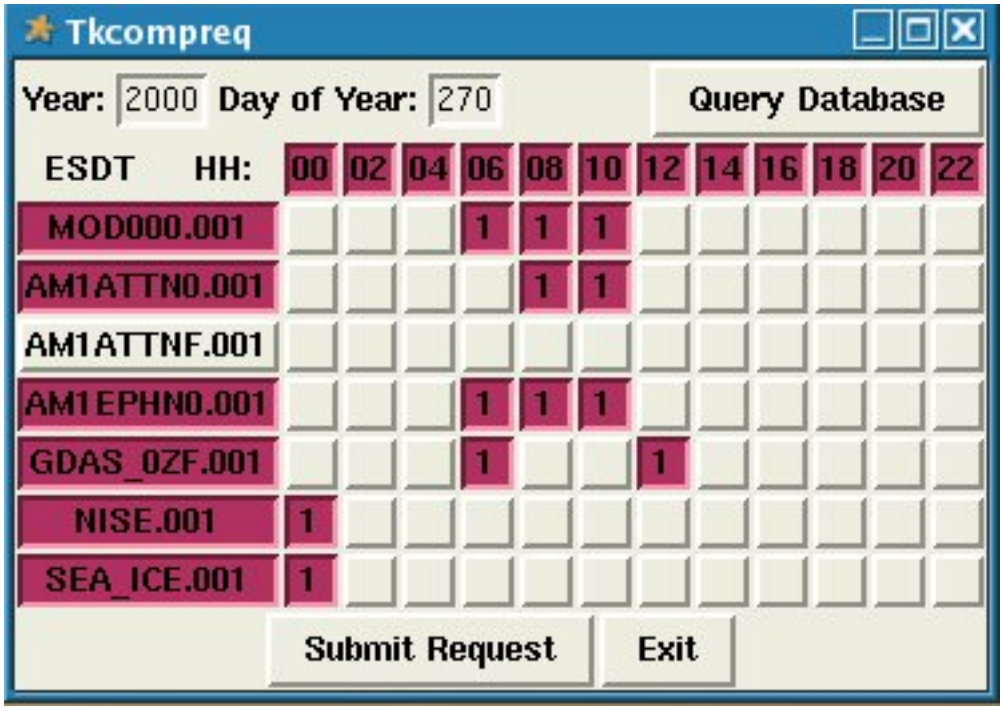

Figure 20-1. The Compose Request tool window. Data types are shown down the left side of the window. To order data, enter the year and day of year in the spaces indicated and then click on Query Database. Once satisfied with order, click on Submit Request.

The Compose Request tool allows a user to order data for up to one full data day. All possible data types (for this S4PM configuration) are listed down the left side. Hours of the day are listed across the top in increments configured when the string was set up (two hours in Figure 20-1).

Ordering data via the Compose Request tool is a two-step process. First, the user needs to query the database for the data available for a particular data day. After the year and day of year are entered, clicking on the **Query Database** will execute a query to the database. Query "hits" are shown as red boxes with the number of files shown within each box. To restrict the query by data type, deselect the data types not desired on the left side. To restrict the query by hour, deselect the hours not desired across the top. A query can be rerun by clicking on the **Query Database** button again. Specific files can be deselected by clicking on the appropriate boxes.

The second step is to submit a request for the data displayed as red boxes by clicking on the **Submit Request** button. Data will be ordered for each of the red boxes shown in the display.

20.2 Station Work Orders

Standard Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	REQUEST_DATA_datatype	PDR	Subscription Notify (Section 26)
	REQUEST_DATA	PDR	Compose Request Tool
Output:	None	N/A	N/A
On-Demand Processing			
Input:	REQUEST_DATA	PDR	Split Services (Section 24)
Output:	REQUEST_DATA	PDR	Track Requests (Section 29)

Table 20-1. Work orders in the Request Data station.

The input work order is a REQUEST_DATA or REQUEST_DATA_datatype in PDR format. In standard real-time processing, the REQUEST_DATA_datatype work order is sent by Subscription Notify whereas the REQUEST_DATA work order (unqualified by data type) is constructed by the Compose Request tool. In on-demand processing, the Split Services station builds the REQUEST_DATA_datatype PDR work order.

In standard processing strings, there is no output work order. Once requested data are received by S4PM, it is the Receive DN (Section 16) station that closes the loop.

In on-demand processing, the only output work order is the REQUEST_DATA work order itself. It is simply re-routed to the Track Requests station as a way of notifying it that a request for data has been made.

20.3 What To Monitor

The Request Data station is single-threaded. That is, only one job is allowed to run in the station at a time. If space for the data to be ordered cannot be allocated, the job should immediately quit and get requeued for another try. Therefore, jobs in this station normally show up only as blue (*i.e.* queued) job.

20.4 What Can Go Wrong

20.4.1 Data Multiplicity

When ordering data via the Compose Request tool (see Section 18.1.1), a query is first made against the ECS science data server. Query "hits" are shown as red boxes with the number of files shown within each box. A '1' means that only one file exists for that data type and data time. A '2' would mean that there are two files matching those criteria. If the number in any red box is *greater than one* when you click the **Submit Request** button, the request will fail. The reason is that if there is more than one file matching the specified criteria, the tool cannot make the decision as to which file to actually order.

When there is a failure for the above reason, the only recourse is to have duplicate file(s) marked as deleted from the ECS archive leaving only one file matching the given data type and data time. This means that a decision will have to be made as to what files should be deleted and what one file should remain. This is beyond the scope of this document.

20.4.2 Job Fails

A job may fail in the Request Data station if it fails to invoke the ECS SCLI tool. To see if this was the cause of the failure, view the SCLI.log file in the Request Data station directory. Note that if the SCLI is being invoked remotely on another box, you will need to locate the SCLI.log file on that other box and view it. You can tell if SCLI is being invoked locally or remotely by examining the station.cfg file in the Request Data station. If you see the script s4pm_request_data.pl being called with the –h <machine> argument, it means that the SCLI is being invoked remotely on the machine so named. If there is no –h argument, the SCLI is assumed local.

20.4.3 Job Stays Blue In The Queue

From time to time, you may notice jobs never seem to be able to find the disk space they need and are continuously being recycled. In this case, you need to examine the chain log file for one or more of these jobs. There, you will see which disk pool is too full. You can also see this in the View Disk Allocation and Usage Tool (Section 8.3).

In any case like this, there are several options:

1. Verify in the chain log that the disk pool in which space is being sought exists and is reasonably sized for the string. If not, you may need to reconfigure disk pool allocations (see Section 8.2). If jobs are looking for space in a disk pool that doesn’t exist, there is a configuration inconsistency in the string and the string may need to be reconfigured.
2. If space is unavailable due to some transient backlog (for example, some data type was delayed a long time in getting to S4PM), you may just want to be patient.

Alternatively, you may want to open up the size of the pertinent disk pool temporarily until throughput is restored.

21. The Run Algorithm Station

21.1 Station Overview

Is this a component …	Of this S4PM flavor?
✓	Standard Real-Time Processing
✓	Standard Retrospective Processing
✓	On-Demand Processing
✓	Data Mining
✓	With Datapool Polling
✓	With Insert to Datapool

The Run Algorithm station is responsible for running the algorithms. Any data produced by the algorithm are registered within S4PM so that other algorithms that use those data as input can be triggered. Further, Run Algorithm updates the Track Data station on the data that were used by the algorithm. The Track Data station needs this information so that it knows when it can have the data safely deleted. Run Algorithm also generates a Production History tar file (unless that option has been turned off). Also, Run Algorithm gathers statistics on each run of an algorithm. These statistics are rolled up into text files that provide information on the number of algorithms run, the volume of data produced, etc. per unit time.

The Run Algorithm makes this assumption about the exit codes from all algorithms it runs: an exit code of zero means success and any other exit code means failure. If an algorithm fails in Run Algorithm, the station will construct a Failed PGE tar file. This tar file contains the log files along with other informational files from the algorithm run. The Failed PGE tar file can later be used to investigate the failure. If, after a failure, the **Punt** button is clicked in the S4PM Job Monitor window, the Failed PGE tar file is exported to the ECS archive. In On-Demand processing, there is instead the **Fail The Order** which, although it doesn't try to export the Failed PGE tar file to the archive, will recover any allocated disk space.

21.2 Station Work Orders

Standard Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	RUN_algorithm	PCF	Allocate Disk (Section 8)
Output:	CLEAN	Pathname	Sweep Data (Section 27)
	EXPORT	PDR	Export (Section 11)
	TRIGGER_DATA	PDR	Register Local Data
	UPDATE	Pathname	Track Data (Section 28)
On-Demand Processing			
Input:	RUN_service	PCF	Allocate Disk (Section 8)
Output:	CLEAN	Pathname	Sweep Data (Section 27)
	EXPORT	PDR	Export (Section 11)
	TRIGGER_DATA	PDR	Register Local Data
	UPDATE	Pathname	Track Data (Section 28)

Table 21-1. Work orders in the Run Algorithm station.

21.3 What Can Go Wrong

Most failures in the Run Algorithm station are due to failures in the algorithms themselves. If there as a failure, first look at the chain log file (RUN_*.log). Another place to look is the LogStatus file if the algorithm is using the ECS Toolkit. If this file does not exist, it means either that the algorithm isn't using the ECS Toolkit or the failure occurred before the LogStatus file was created. If the latter is the case, then the cause of the failure is more likely to be station configuration or a problem in the input RUN work order.

22. The Select Data Station

22.1 Station Overview

Is this a component …	Of this S4PM flavor?
☑	Standard Real-Time Processing
☑	Standard Retrospective Processing
☐	On-Demand Processing
☑	Data Mining
☑	With Datapool Polling
☑	With Insert to Datapool

In standard processing S4PM strings, the Select Data station is responsible for determining what input data are needed for a particular algorithm run. The station is driven by configuration files, one per algorithm, that embody the production rules for that algorithm.

When particular data arrive in S4PM (either from an external source or produced within S4PM), the Select Data station receives notification of those new data via a NEWDATA_algorithm work order. Based upon the characteristics of the data (usually time coverage and data type) and the production rules for that algorithm, Select Data determines what other data are needed in order for that algorithm to run. Both required and optional data are determined.

The production rules for each algorithm are specified in algorithm-specific configuration files named s4pm_select_data_algorithm.cfg and they reside in the select_data_cfg subdirectory under the station directory.

The Select Data station is not a component of on-demand processing. It should be noted, however, that the production rule configuration files are nonetheless needed in on-demand processing and they reside in the Split Services (Section 24) station directory.

22.2 Station Work Orders

Standard Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	NEWDATA_algorithm	PDR	Register Data (Section 18)
			Register Local Data (Section 18)
Output:	MOREDATA_algorithm	PDR	Find Data (Section 12)

Table 22-1. Work orders in the Select Data station.

The input work order to Select Data is a NEWDATA_algorithm work order either from Register Data or Register Local Data depending on whether the data were from external sources or produced within S4PM. The format is PDR. Each algorithm has a single data type whose arrival in S4PM triggers its running. This data type is called the trigger data type. The NEWDATA work order contains only one data file whose data type is a trigger data type for the algorithm so named in the work order file name.

The output work order is the MOREDATA_algorithm, again in PDR format. In addition to the trigger data type from the input work order, the output work order contains references to other data required for a single run of this algorithm. In addition to containing the required data references, it also contains all possible optional data references. The data file references in the output PDR will become actual data file names and directories once the Find Data station has located the data.

23. The Ship Data Station

23.1 Station Overview

Is this a component …	Of this S4PM flavor?
☐	Standard Real-Time Processing
☐	Standard Retrospective Processing
☑	On-Demand Processing
☐	Data Mining
☐	With Datapool Polling
☐	With Insert to Datapool

In on-demand processing S4PM strings, the role of Ship Data is to ship data processed on behalf of a user's request to that user. The Ship Data receives a SHIP work order from the Track Requests (Section 29) station when an order has been completed. Using the information contained in the SHIP work order, the station obtains an ECS order ID and request ID from the MSS database and using these IDs, submits a distribution request via the Distribution Command Line Interface (DCLI), an ECS-provided tool. The DCLI then routes the data to the user using the distribution services of ECS.

Note that as of release 5.8.1, the preferred method of distribution is ECD PDR Distribution (EPD). This is the default mechanism for all orders coming in through the V0 Data Gateway. For such orders, the Ship Data station is not used. Instead, distributions are handled via the Export station. The Ship Data station is only used for orders not coming in via theV0 Data Gateway (*e.g.* orders through the WHOM interfaces at the GES DISC).

23.2 Station Work Orders

On-Demand Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	SHIP	ODL	Track Requests (Section 29)
Output:	CLOSE	PDR	Track Requests (Section 29)
	CLOSE	PDR	Track Data (Section 28)

Table 23-1. Work orders in the Ship Data station.

24. The Split Services Station

24.1 Station Overview

Is this a component …	Of this S4PM flavor?
☐	Standard Real-Time Processing
☐	Standard Retrospective Processing
☑	On-Demand Processing
☐	Data Mining
☐	With Datapool Polling
☐	With Insert to Datapool

The Split Services station is part of on-demand processing only. Its main job is to accept ODL requests from WHOM (and soon, EDG) and break it up into separate orders according to media type (DVD, FTP push, FTP pull). The ODL arrives at Split Services as a SERVICE work order and it contains the specific data on which one or more services are to act. A typical service is subsetting. In addition, the SERVICE ODL work order contains information on the user requesting the service so that the data resulting from running the services can be distributed to him or her.

24.2 Station Work Orders

On-Demand Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	SERVICE	ODL	EDG, Other Client
Output:	MOREDATA_service	PDR	Find Data (Section 12)
	TRACK_REQUEST	PDR	Track Requests (Section 29)
	ORDER_FAILURE	PDR	Track Requests (Section 29)
	REQUEST_DATA	PDR	Request Data (Section 20)
	EXPECT	Pathname	Track Data (Section 28)

Table 24-1. Work orders in the Split Services station.

The Split Services station produces an output MOREDATA work order that is sent to the Find Data station. In most S4PM string configurations, it is the job of the Select Data station to produce a MOREDATA work order. But in on-demand processing, the precise data to be requested are already known and included in the input SERVICE work order. Therefore, there is no need for the Select Data station.

Split Services also send a TRACK_REQUEST work order to the Track Requests station to alert it that a new order has been received. Track Requests will then begin tracking that order to completion. In the event of some failure, an ORDER_FAILURE work order is also sent to Track Requests to alert it of the failure.

A REQUEST_DATA work order is sent to the Request Data station so that the data to be serviced can be ordered.

An EXPECT work order is sent to the Track Data station to tell it what input data to expect.

25. The Stage For Pickup Station

25.1 Station Overview

Is this a component …	Of this S4PM flavor?
☐	Standard Real-Time Processing
☐	Standard Retrospective Processing
☐	On-Demand Processing
☑	Data Mining
☐	With Datapool Polling
☐	With Insert to Datapool

The Stage For Pickup station is only used in data mining S4PM strings. The purpose of this station is to move data produced in a data mining string into a FTP pickup area and send a short PAN back to the Receive PAN station upon success.

25.2 Station Work Orders

On-Demand Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	S4PM_*	PDR	Export (Section 11)

Table 25-1. Work orders in the Stage For Pickup station.

The input work order is a PDR whose name depends upon a unique number assigned to the string on behalf of a user (*e.g.* S4PM_DME22). Unlike typical work orders, this one isn't sent to the Stage For Pickup station using Stationmaster. Instead, the Export station is configured to drop this work order into the Stage For Pickup station directory directly.

There is no output work order *per se* from Stage For Pickup. Instead, a PAN is dropped directly into the Receive PAN station upon successful completion of the job.

26. The Subscription Notify Station

26.1 Station Overview

Is this a component …	Of this S4PM flavor?
☑	Standard Real-Time Processing
☐	Standard Retrospective Processing
☐	On-Demand Processing
☑	Data Mining
☐	With Datapool Polling
☐	With Insert to Datapool

The Subscription Notify station is a component of standard real-time processing strings only. It is responsible for converting Insert Notifications into data requests from the ECS. Since Insert Notifications should arrive immediately after data are ingested into ECS, the requests for those same data should be from cache rather than tape in the archive.

The S4PM user running the standard real-time processing strings must be subscribed to Insert Notifications for those data to process, typically Level-0 and ancillary data. The Insert Notifications are sent from ECS via e-mail. These e-mails are parsed by procmail and disseminated to the Subscription Notification station directory (sub_notify).

A major part of the job of Subscription Notification is to create a REQUEST_DATA work order of the data to be requested, those data that are in the Insert Notification e-mail. The resulting REQUEST_DATA work order must then be directed to the S4PM string configured to use those data. To do this, the Subscription Notification station maintains a configuration file mapping data type to the S4PM string that needs it. For each S4PM string needing that data type, Subscription Notification will send a REQUEST_DATA work order to that string's Request Data station (Section 20) so that it can order the data.

Since the Subscription Notification station is a disseminating station, it must physically reside in a single location that is visible to all S4PM strings that need it. For this reason, the Subscription Notification station directory, sub_notify is on a cross-mounted file system. In this way, it is visible from all boxes. A link to this single physical location is

then created in each station root so that it appears next to all other stations on all S4PM strings.

26.2 Station Work Orders

Standard Processing (Real-Time only)			
	Work Order Name	**Format**	**From / To Station**
Input:	Insert Notification	E-Mail	ECS
Output:	REQUEST_DATA_datatype	PDR	Request Data (Section 20)

Table 26-1. Work orders in the Subscription Notify station.

The input work order is the Insert Notification, an e-mail message generated by ECS upon the insert into the archive of subscribed data types. The e-mail is sent to the S4PM user (under whose account S4PM is running) and processed by a procmail filter which directs the e-mail to the Subscription Notify station directory.

The output work order is a REQUEST_DATA_datatype where datatype is the data type in the work order. Output work order names are distinguished by data type so that they can be directed to the Request Data station of the S4PM string needing those data.

27. The Sweep Data Station

27.1 Station Overview

Is this a component …	Of this S4PM flavor?
✓	Standard Real-Time Processing
✓	Standard Retrospective Processing
✓	On-Demand Processing
✓	Data Mining
✓	With Datapool Polling
✓	With Insert to Datapool

The Sweep Data station is responsible for physically deleting data no longer needed in S4PM. The station will also deallocate any disk space reserved for the deleted data and remove any data request stub files which the Request Data uses to track which data were already requested and which were not.

27.2 Station Work Orders

Standard Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	CLEAN	Pathname	Run Algorithm (Section 21)
			Track Data (Section 28)
Output:	None	N/A	N/A
On-Demand Processing			
Input:	CLEAN	Pathname	Run Algorithm (Section 21)
			Track Data (Section 28)
Output:	None	N/A	N/A

Table 27-1. Work orders in the Sweep Data station.

The input work order for both standard and on-demand processing is the CLEAN work order. The CLEAN work order is in the pathname format, a simple list of the full pathnames of the data to sweep away.

There is no output work order from the Sweep Data station.

28. The Track Data Station

28.1 Station Overview

Is this a component …	Of this S4PM flavor?
☑	Standard Real-Time Processing
☑	Standard Retrospective Processing
☑	On-Demand Processing
☑	Data Mining
☑	With Datapool Polling
☑	With Insert to Datapool

The job of Track Data is to keep track of all data within S4PM. When data are no longer needed, Track Data is responsible for initiating their deletion.

Track Data uses the Perl DB database module to maintain two database files: uses.db and path.db. The uses.db DB file maps data files currently within S4PM to the current number of uses outstanding against those files. The path.db DB file maps those same data files to their directory locations.

The uses.db is particularly important. When a data file is first registered within S4PM either via Register Data or Register Local Data (Section 18), it is added to both the uses.db and path.db databases. The number of uses assigned to the file is determined by a prediction of how many times the data file will be used before it can be safely deleted. A use means either being read by something (*e.g.* a service or algorithm) or a distribution (via Export or Ship Data). Every time a use is made against a data file, the uses in the uses.db file is decremented. When the number of uses reaches zero, Track Data knows that the file can be deleted from the S4PM system.

Since not all files are used the predicted number of times, there needs to be a safety net mechanism for having those files deleted since their uses will never reach zero. This too is handled in S4PM. One of the jobs running in the Repeat Hourly station (Section 19) examines the time stamps on all data files. When a data file ages beyond a configurable limit, a work order is sent to Track Data to zero out that data file's uses. As with any other file whose uses becomes zero, the file will then be deleted.

28.1.1 Delete Data Tool

The Delete Data Tool provides a way to properly delete data files from the S4PM system. It handles the physical deletion of the data and associated files as well as the deallocation of space reserved in the disk pools. The tool is available by right-clicking on the Track Data station button in the S4PM Monitor window. An example is shown in Figure 25-1.

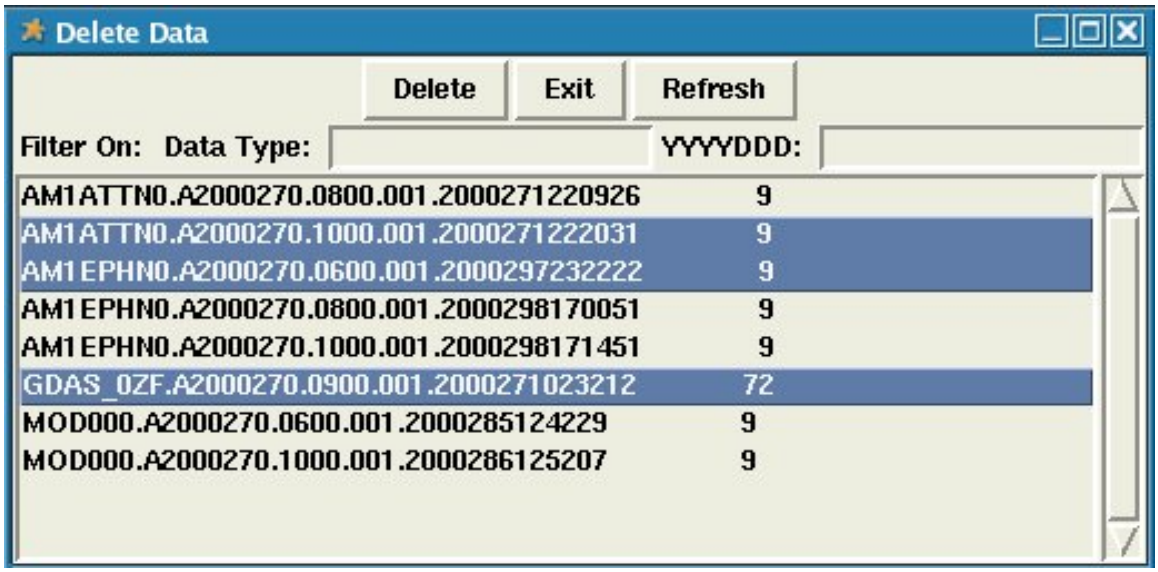

Figure 28-1. Image of the Delete Data tool. The items highlighted in blue will be deleted once the Delete button is clicked.

The Delete Data tool is a simple tool that displays the files currently resident in S4PM along with their outstanding uses. To use the tool, simply select one or more files in the window and click on **Delete**. The **Refresh** button allows the display to be refreshed (if any new files came into S4PM while running this tool). When you select **Delete**, a pop-up box will confirm your selection before acting on it.

To deal with large numbers of files resident in S4PM, a rudimentary filter mechanism is available by data type or data date.

28.2 Station Work Orders

Standard Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	UPDATE	Pathname	Receive PAN (Section 17), Run Algorithm (Section 21), Repeat Hourly (Section 19)
	INSERT	Pathname	Register Data, Register Local Data (Section 18)
Output:	CLEAN	Pathname	Sweep Data (Section 27)
On-Demand Processing			
Input:	UPDATE	Pathname	Ship Data (Section 23), Run Algorithm (Section 21), Repeat Hourly (Section 19)
	INSERT	Pathname	Register Data, Register Local Data (Section 18)
	EXPECT	Pathname	Split Services
Output:	CLEAN	Pathname	Sweep Data (Section 27)

Table 28-1. Work orders in the Track Data station.

In both standard and on-demand processing, the INSERT work order is used for having Track Data add an entry into its databases for a new data file. The work order format is full pathname of the data file and the initial (predicted) number of uses with which to associate it. The UPDATE work order is for data that already exist in the Track Data databases. The format is similar to an INSERT work order, but rather than containing the number of uses, it instead contains the number of uses to subtract. Track Data takes that number and uses it to decrement the number of uses for that data file.

If the number of uses for a data file falls to zero or below, Track Data will delete the relevant entries from its database files and send a CLEAN work order to the Sweep Data (Section 27) station that will carry out the physical deletion of the file.

In on-demand processing only, an EXPECT work order is sent to Track Data by Split Services (Section 24) to alert it of data that is "on the way". This allows Track Data to maintain a placeholder in its databases for data it expects to arrive.

28.3 What To Monitor

As mentioned earlier, jobs in the Allocate Disk station run very quickly and job failures should be extremely rare.

To monitor the current state of the disk pools, use the View Disk Allocations and Usage tool. Each disk pool is represented in the window with the pool name on the far left. The bar to the right indicates the fraction of the available space in the disk pool that is currently allocated. The three columns on the right side quantitatively list the number of data granules in the pool, the number of additional granules that can be allocated, and the total number of granules that can be allocated. Click on the Refresh button to update the window.

28.4 What Can Go Wrong

As mentioned above, jobs that cannot get disk space allocated are immediately recycled. Typically, space will be available the second or third time through. If a disk pool is full, however, the number of times a work order is cycled can grow large. Usually, disk pools filled to capacity is a transient even and should raise no concern.

Sometimes, due to misconfiguration, a job may try to request space in a disk pool that doesn't exist. In this case, the requested space is "never" available and the work order is recycled indefinitely. To check for such occurrences if you suspect them, examine the work order log files. If you see something like:

29. The Track Requests Station

29.1 Station Overview

Is this a component …	Of this S4PM flavor?
☐	Standard Real-Time Processing
☐	Standard Retrospective Processing
☑	On-Demand Processing
☐	Data Mining
☐	With Datapool Polling
☐	With Insert to Datapool

The Track Requests station is only used in on-demand processing. Its responsibility is to track a user's order, which may have been broken up into multiple requests, to completion. Once Track Requests determines that an order is complete, it sends a SHIP work order to the Ship Data station so that the data can be distributed to the user.

If there is a failure in processing a user's order, Track Requests will receive an ORDER_FAILURE work order from the station where the failure occurred. If at least one of the data products of an order was successfully produced, the Track Requests will nonetheless succeed and the user will receive the data that was successfully produced along with a notification that a portion of the order failed. The user may then try again or contact User Services.

If the user's order fails completely, the job in Track Requests will fail. Clicking on the Ignore Order Failure button in the S4PM Job Monitor window will clean up the debris and notify the user that the entire order failed.

29.2 Station Work Orders

On-Demand Processing			
	Work Order Name	**Format**	**From / To Station**
Input:	TRACK_REQUEST	ODL	Split Services (Section 24)
	TRIGGER_service	PDR	Find Data (Section 12)
	GETDISK_service	PCF	Prepare Run (Section 14)
	RUN_service	PCF	Prepare Run (Section 14)
	EXPORT	Pathname	Run Algorithm (Section 21)
	CLOSE	ODL	Ship Data (Section 23)
	PAN	E-Mail	Export (Section 11)
	ORDER_FAILURE	Mixed	Multiple Stations
Output:	SHIP	PDR	Ship Data (Section 23)

Table 29-1. Work orders in the Track Requests station.

The main input work order is the TRACK_REQUESTS work order from the Split Services station to alert it that a new order has come into the system.

Between the time when Track Requests is alerted to a new order and the time that it sends out a SHIP work order, the station tracks the progress of the order through various stages. It does so by receiving work orders from many stations involved with processing the order. Track Requests receives such tracking work orders from Split Services, Request Data, Find Data, Prepare Run, Allocate Disk, Run Algorithm, and Ship Data. None of these work orders are read by Track Requests; they are simply copies of work orders sent among the other stations and serve only to let Track Requests know the progress of an order.

Appendix A. Acronyms

Acronym	Meaning
AIRS	Atmospheric Infrared Sounder
ASCII	American Standard Code for Information Interchange
DAAC	Distributed Active Archive Center
DN	Distribution Notification
ECS	EOSDIS Core System
FTP	File Transfer Protocol
EOSDIS	Earth Observing System Data Information System
GDAAC	GES Distributed Active Archive Center
GES	Goddard Earth Sciences
GSFC	Goddard Space Flight Center
GUI	Graphical User Interface
HDF	Hierarchical Data Format
LGID	Local Granule ID
LUN	Logical Unit Number
MODIS	Moderate Resolution Imaging Spectroradiometer
ODL	Object Data Language
OS	Operating System
PAN	Product Acceptance Notification
PCF	Process Control File

PGE	Product Generation Executive, another name for algorithm
PH	Production History
PDR	Product Delivery Record
PDPS	Planning and Data Processing Subsystem
PDRD	Product Delivery Record Discrepancy
QA	Quality Assessment
QC	Quality Control
S4P	Simple, Scalable, Script-Based, Science Processor
S4PM	Simple, Scalable, Script-Based, Science Processor for Missions
SCLI	SDSRV Command Line Interface
SDSRV	Science Data Server
SIPS	Science Investigator-Led Processing System
UR	Universal Reference