Science Software Integration and Test

July 2006

Overview of Lesson

- Introduction
- Science Software Configuration Management
- Science Software Integration and Test (SSI&T) Preparation, Processes and Setup
- Science Software Integration and Test (SSIT) Manager
- Acquiring and Unpacking the Delivered Algorithm Package (DAP)
- DAP Acquire
- Inserting a Science Software Archive Package (SSAP)
- Updating a Science Software Archive Package (SSAP)

Overview of Lesson (Cont.)

- **Standards Checking of Science Software**
- **Compiling and Linking Science Software**
- **Building Science Software with the SCF Version of the SDP Toolkit**
- **Running a PGE in a Simulated SCF Environment**
- **The ECS Assistant Functionality Replaced in Part by Scripts and Monitor GUI Whazzup**
- **ESDT Management**
- **Production Rules**
- **Data Preprocessing (DPREP)**

Overview of Lesson (Cont.)

- **Updating the Orbit Model**
- **PGE Registration and Test Data Preparation**
- **PGE Planning, Processing, and Product Retrieval**
- **Postprocessing and General Investigation**
- **File Comparison and Data Visualization**
- **Miscellaneous**
- **Practical Exercise**

Science Software Integration and Test (SSI&T) Overview

- **WHAT SSI&T IS**
  - **SSI&T is the process by which science software developed by Instrument Teams at local SCFs is tested and integrated into the production system at the DAACs**
  - **The scope of SSI&T for pre-launch releases covers activities starting with delivery of the science software to the DAACs and ending with either the successful integration of each delivered PGE into the production system or the scheduled end date for SSI&T support**
  - **SSI&T is a team effort which can only be successful in the allotted time if all groups cooperate**

Science Software Integration and Test (SSI&T) Overview (Cont.)

- **WHAT SSI&T IS NOT**
  - **SSI&T is not validation of science algorithms that are incorporated into PGEs which produce science data products**
  - **SSI&T is not validation of the science data produced**
  - **Although the Operational Procedures are written as checklists of menu driven activities, SSI&T is not a turnkey process which can be run by test personnel who have no knowledge and experience related to science software development and data processing**
  - **SSI&T is not a simulation of production**

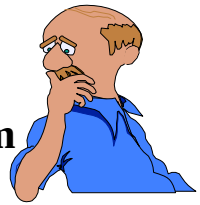
SSI&T Process Overview - The Big Picture

1

Science Software Received

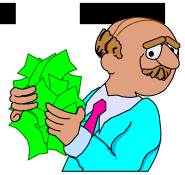
2

SSIT Team



- Receive and Inspect New PGE and Associated Software which includes:
  - •PGE Source Code & Status Messages
  - •Make File
  - •MCFs for Output files for each ESDT
  - •Shell Scripts and Instructions
  - •Process Control files (PCF).
  - •Input/ Output ESDT Descriptor files
  - •Input Test Granules

3



Configuration Manager

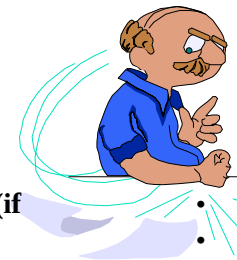
Receives and controls using ClearCase

4

SSIT Team

Pre-SSIT

- Prepare ESDTs (if needed)
- Perform Standards Checking on all files
- Make changes
- Compile/Link PGEs to the SCF Toolkit
- Run and Profile PGE
- Perform file comparison and problem tracking.



Formal SSIT

- Install ESDTs/ODLs
- Register PGEs in PDPS Database
- Install Input Files in Data Server
- Build PGE with DAAC Toolkit
- Run PGE in PDPS
- Examine products and output log files

5

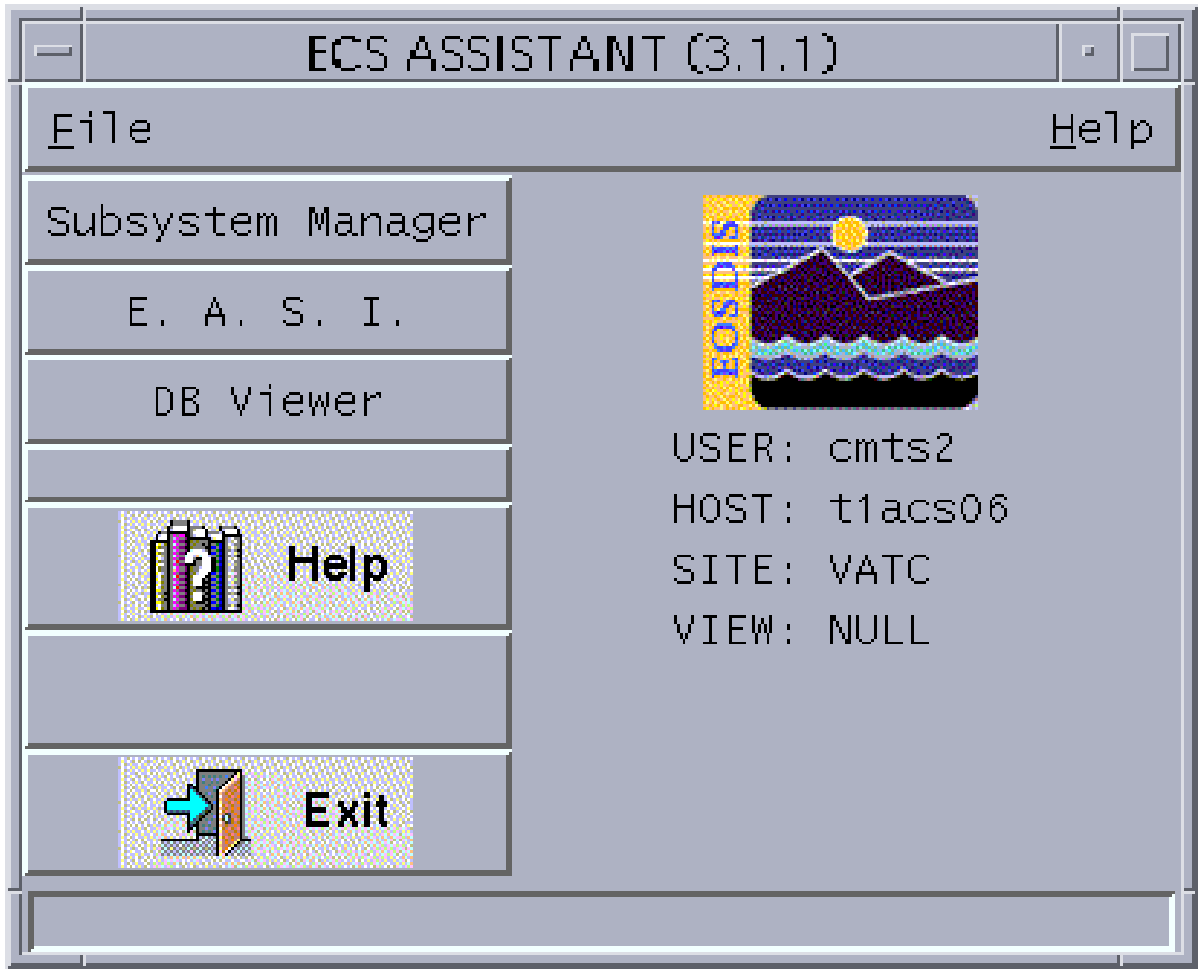
Planning and Processing Team

- Register a Subscription for Test Input/Output files
- Submit a Production Request
- Create a new Production Plan
- Activate the Plan
- Monitor Production

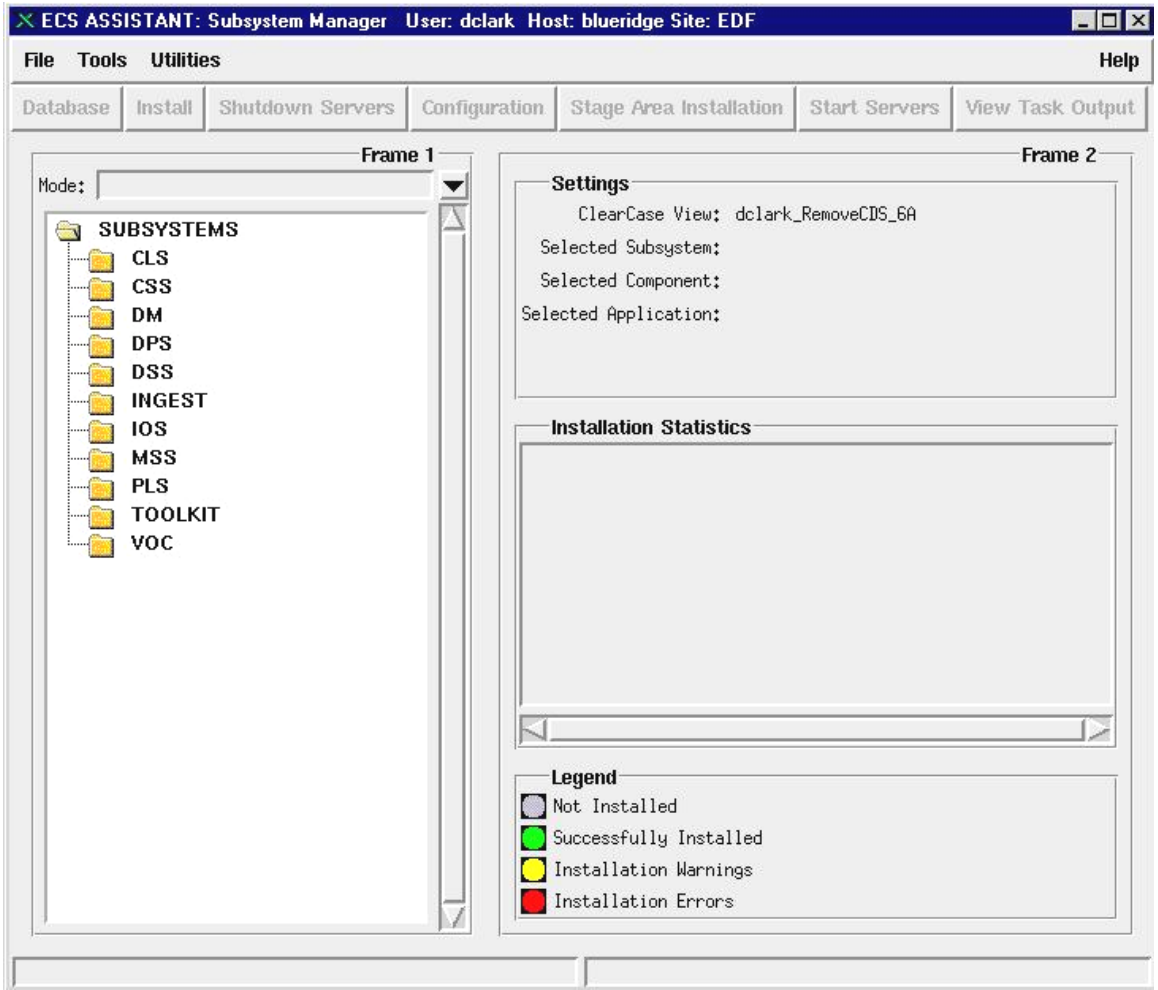
The ECS Assistant

- **Using ECS Assistant to Perform Subsystem Management Functions**
- **Using ECS Assistant to View Databases**
- **Using ECS Assistant to Perform Installs**

ECS Assistant GUI



Subsystem Manager GUI



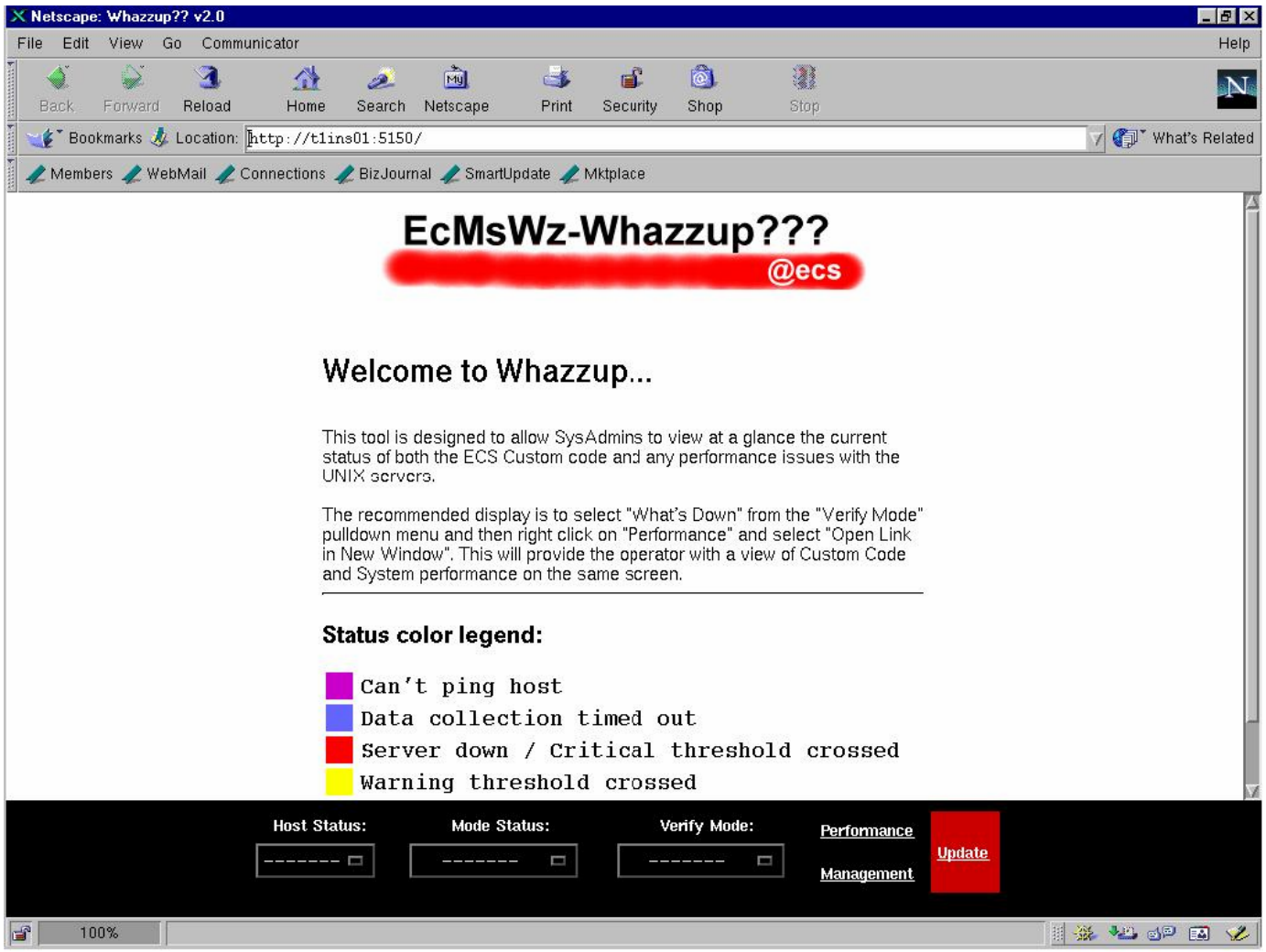
ECS Assistant, Scripts and Monitor GUI Whazzup

- **Currently, the ECS Assistant has lost its capability to monitor the system**
- **This capability now exists with GUI Whazzup**
- **ECS Assistant continues to have the ability to monitor subsystem functions and is mainly used for doing ECS Assistant Subsystem Installs (E.A.S.I.) and staging ESDT/DLL's into the directories, CUSTOM/data/ESS and CUSTOM/lib/ESS respectively**
- **Database Review capability still exists however**
- **The ESDT Manager installation/deletion functions are no longer available nor is the ability to start up and shut down subsystem servers**

Scripts and Monitor GUI Whazzup/ISQL Browsers

- **SSI&T operators can use the following scripts to perform the specified functions:**
  - **Start up or shut down all servers at the same time, a script is used that accesses a list of subsystem servers**
  - **Start up and shut down servers individually using a script established within each subsystem**
  - **Graphically monitor the server up/down status with the Whazzup GUI**
  - **View ESDTs for SSI&T**
  - **Review various databases used in the system by using an ISQL Browser established in each subsystem**

Server Monitor WHAZZUP



Preparation and Setup

- **Preparation for SSI&T is a cooperative effort by Operations and SSI&T**
- **Operations Activities for Preparation and Setup:**
  - **Setup of user accounts**
  - **Setup ClearCase VOB**
  - **Installation of Software Drops**
- **SSIT Preparation and Setup:**
  - **Examine .cshrc file**
  - **Set additional environmental variables**
  - **Source additional setup files**
  - **Verify access to ClearCase VOB**
  - **Procedures are ordered in a logical sequence**
  - **Some deviation will be required for actual work at each DAAC**

Science Software Configuration Management

- **ClearCase - COTS tool for configuration management of science software**
- **Invocation Methods - Command line or Graphical User Interface (GUI)**
- **Key Terms -**
  - **Versioned Object Base (VOB) - a mountable file system which stores version controlled data in directories and files**
    - **Usually accessed with standard UNIX and ClearCase Tools**
  - **View - A working context for a user. Used to access any VOB to make files and directories visible and accessible. Comprised of a storage area for checked out files**
  - **Element - File or directory in ClearCase VOB**

Creating and Setting a View

- **Naming Conventions**
  - **Provides file/directory names for locating directories or files**
  - **Key Names**
    - **ViewName - name of the user's view**
    - **PathName - pathname is the path to the VOB directory**
- **Scope**
  - **Needs to be created only once**
  - **Must be set at beginning of each user session**
- **Key Assumptions**
  - **ClearCase is available**
  - **A VOB has been created**

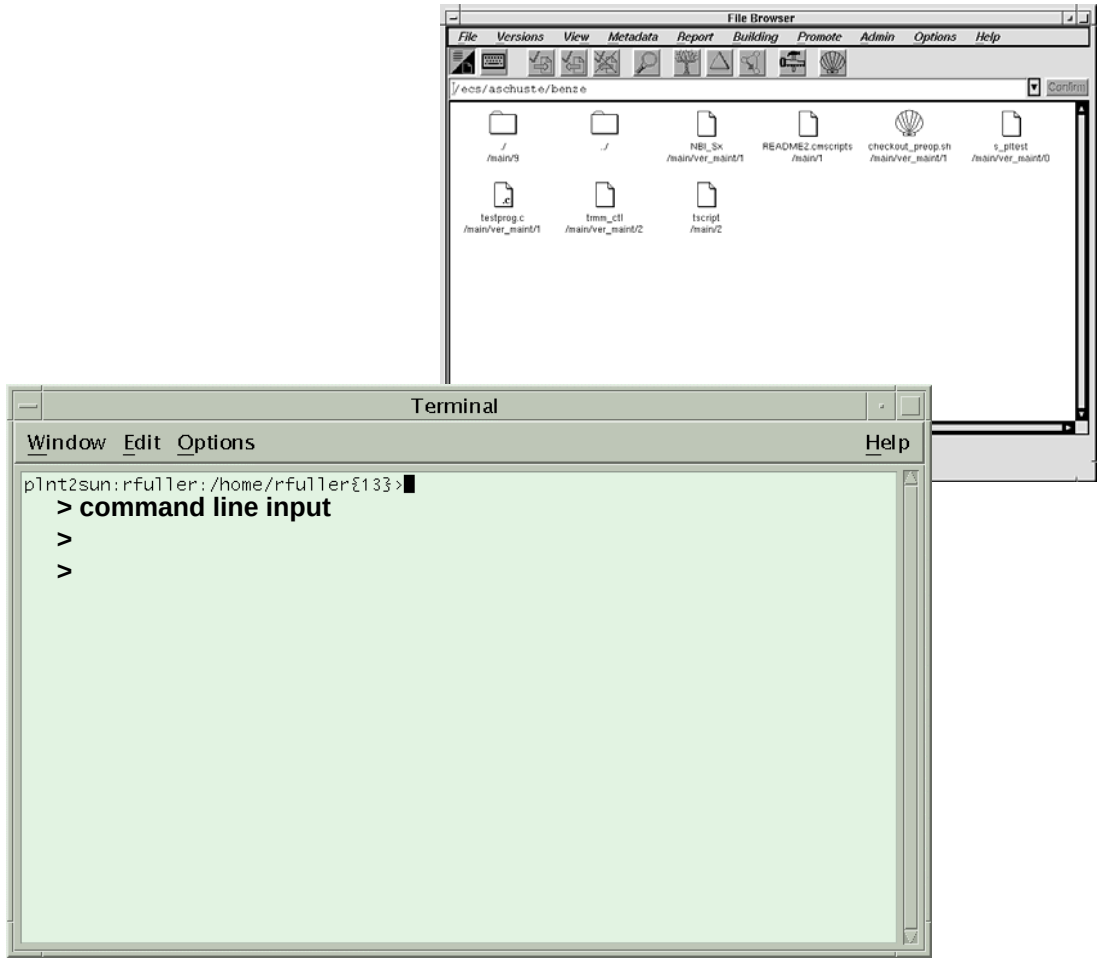
Creating a View

Access ClearCase by typing:
cleartool

Create a view by typing:
mkview -tag *ViewName*
ViewPath/ViewName.vws

View is created named *ViewName*

***ViewName* = name of view**
***ViewPath* = path to view directory**
vws = file extension



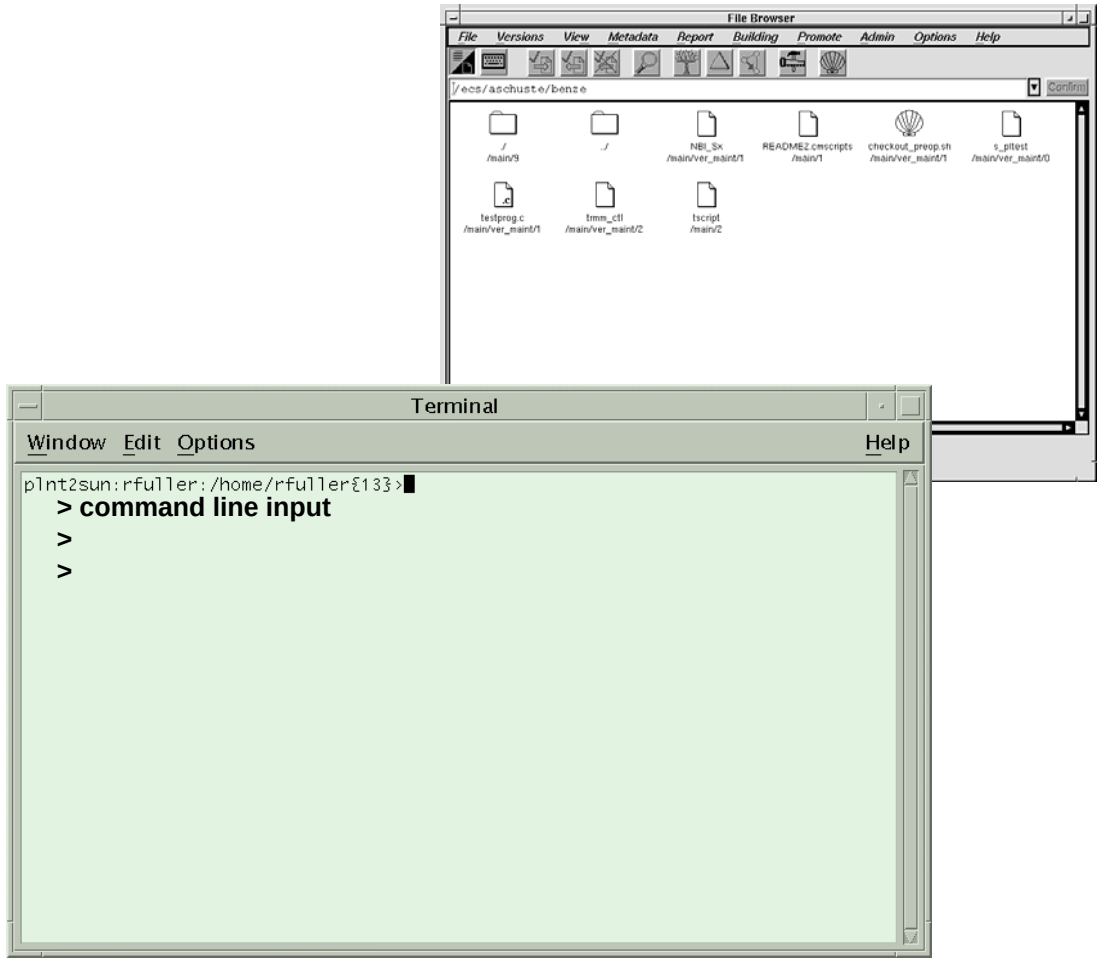
Setting a View

Access ClearCase by typing:
cleartool

Set a view by typing:
setview *ViewName*

View is set to *ViewName*

***ViewName* = name of view**



Entering a New Directory into ClearCase

- **Key Assumptions**
  - **A VOB and subdirectory has been created to hold the file**
  - **A view has been created**
- **ClearCase Commands**
  - **cleartool setview *ViewName* - Launches ClearCase and displays the user's view**
  - **cd *PathName* - Changes directory to a subdirectory in the VOB**
  - **cleartool checkout -nc . - Checks out a directory from ClearCase**
  - **cleartool mkdir -nc *DirName* - Creates a new directory (subdirectory)**
  - **cleartool checkin -nc *DirName* - Checks new directory into ClearCase**
  - **cleartool checkin -nc . - Checks the current directory into ClearCase**

Creating a Directory

Access ClearCase by typing:
cleartool setview *ViewName*

↓

Change Directories by typing:
cd *PathName*

↓

Checkout directory by typing:
cleartool checkout -nc .

↓

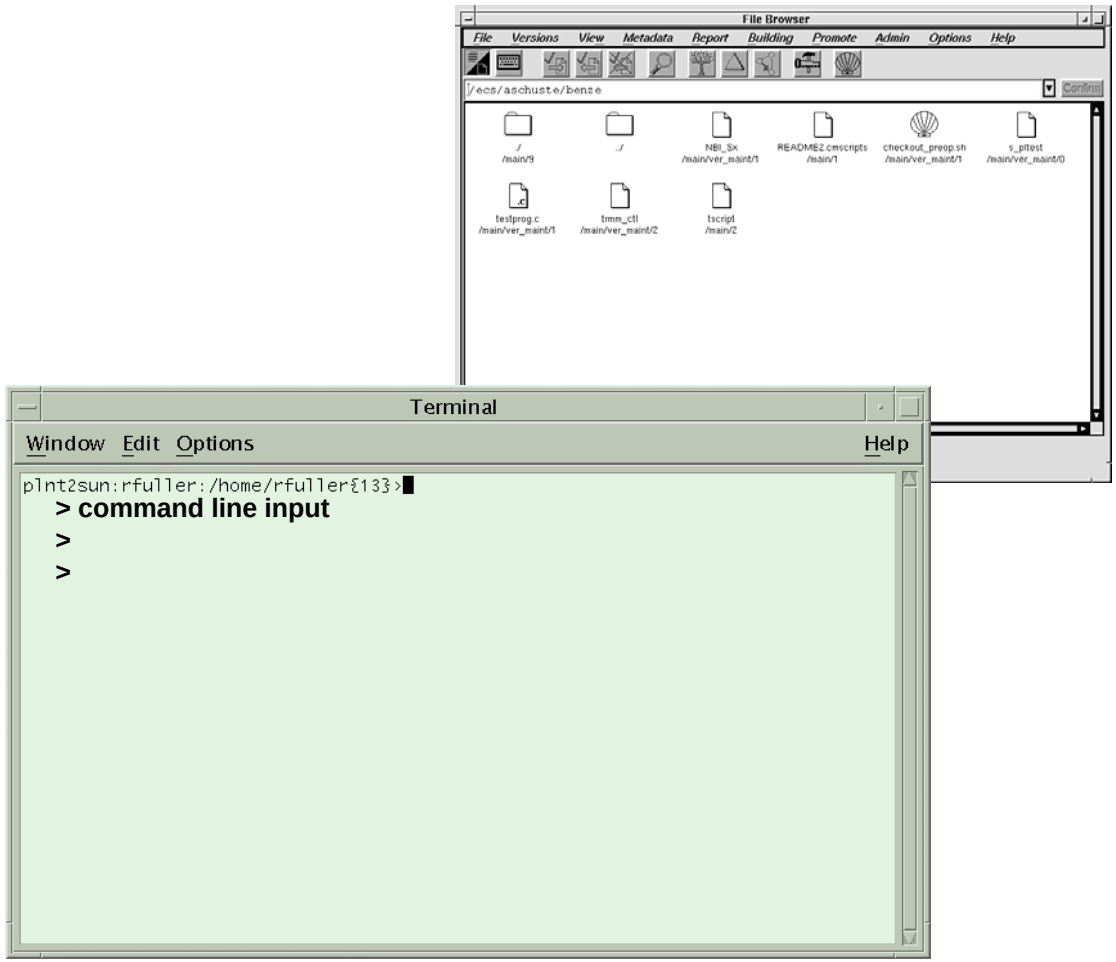
Create a new directory by typing:
cleartool mkdir - nc *DirName*

↓

Checkin a directory by typing:
cleartool checkin -nc *DirName*

↓

A new directory has been created and checked into ClearCase.



Entering a Single File into ClearCase

- **Key Assumptions**
  - **A VOB and subdirectory has been created to hold the file**
  - **A view has been created**

Entering a Single File into ClearCase (Cont.)

- **ClearCase Commands**
  - **cleartool setview *ViewName* - Launches ClearCase and displays the user's view**
  - **cd *PathName* - Changes directory to a subdirectory in the VOB**
  - **cp *FilePath*/*FileName* . - Copies file from current to VOB directory**
  - **cleartool checkout -nc . - Checks out the current directory**
  - **cleartool mkelem -nc *FileName* - Creates a new element/file**
  - **cleartool checkin -nc *FileName* - Checks the file into ClearCase**
  - **cleartool checkin -nc . - Checks the current directory into ClearCase**

Entering a Single File into ClearCase

Access ClearCase by typing:
cleartool setview *ViewName*

↓

Change Directories by typing:
cd *Pathname*

↓

Checkout directory by typing:
cleartool checkout -nc .

↓

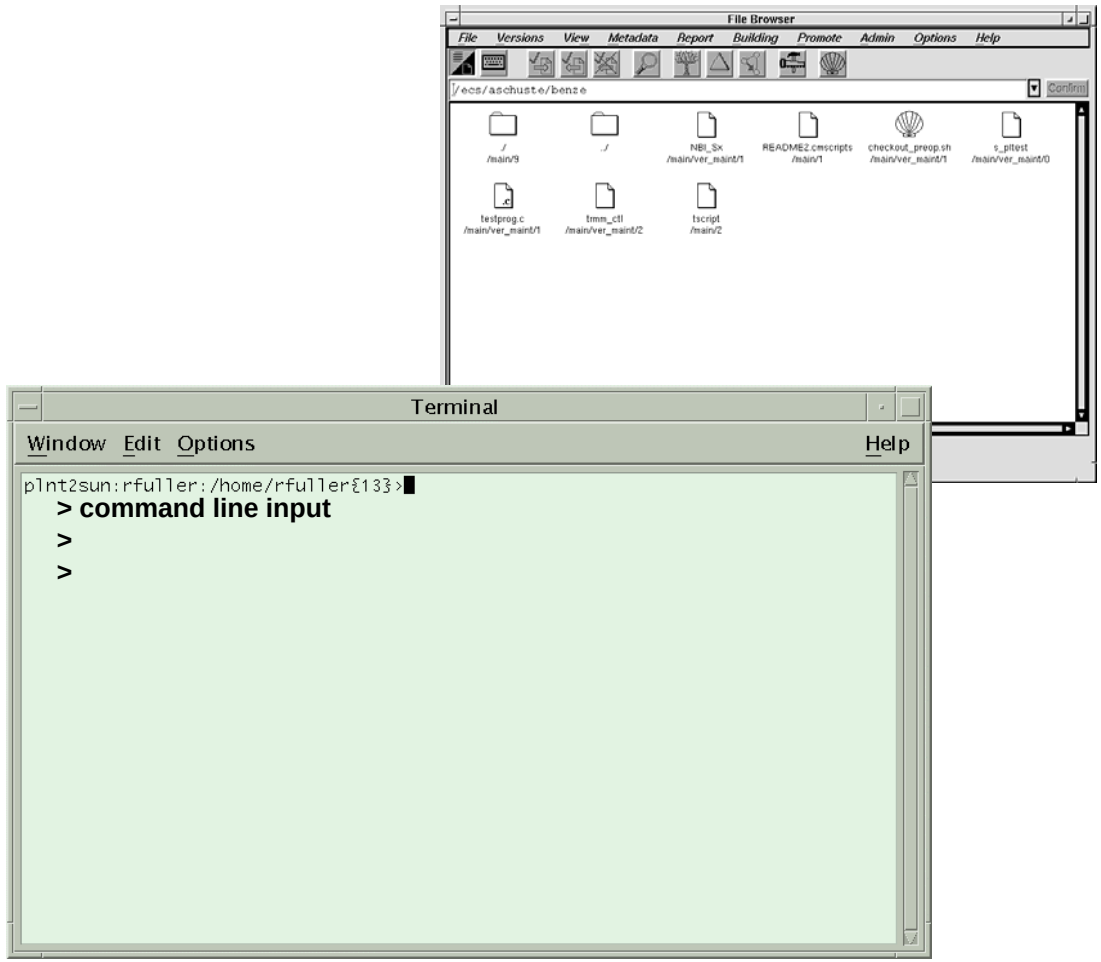
Create a new element by typing:
cleartool mkelem -nc *FileName*

↓

Checkin a file by typing:
cleartool checkin -nc *FileName*

↓

Checkin a directory by typing:
cleartool checkin -nc .



Importing Files into ClearCase

- **Key Assumptions**
  - **DAAC SA required to complete this procedure.**
  - **A VOB and subdirectory has been created to hold these files.**
  - **No object files or executables exist in the directory.**
  - **The PGE was received with a directory structure that contains various types of files.**
  - **The PGE directory structure will be maintained.**

Importing Files into ClearCase (Cont.)

- **ClearCase Commands**
  - **cd *ParentDir* - Changes directory to the parent directory of the directory structure to be brought into ClearCase.**
  - **clearcvt-unix -r *DirName* - Creates a conversion script to import everything in *DirName* directory and everything below it to ClearCase.**
  - **cvt_script - Name of created script and command to run the script to place all elements under ClearCase.**

Importing Multiple Files into ClearCase

Change directories to parent of directory structure:
cd *ParentDir*

↓

Create a conversion script :
clearcvt-unix -r DirName

↓

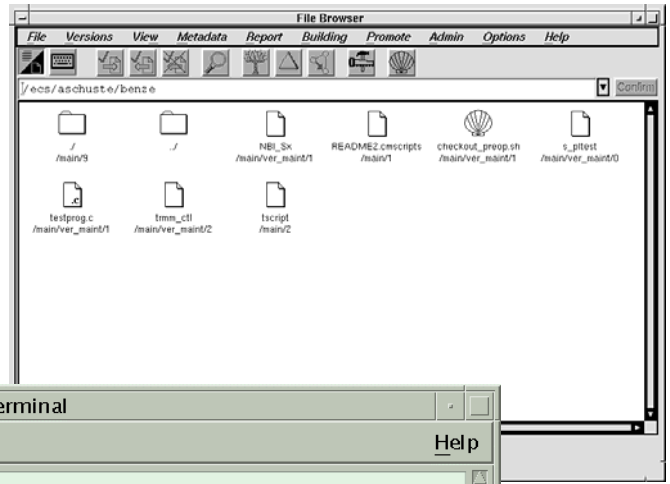
The output script is named:
cvt_script

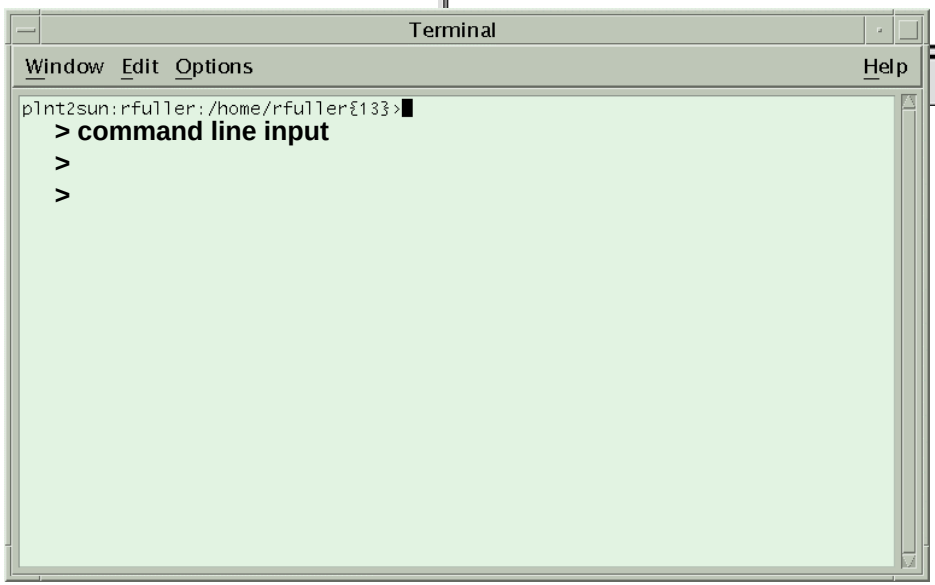
↓

Change directories in VOB by typing:
cd *Pathname*

↓

Copy files into ClearCase by typing the script: **cvt_script**





Checking Out an Element from ClearCase

- **Key Assumptions**
  - **A VOB and subdirectory has been created to hold the file.**
  - **A view has been created.**
- **ClearCase Commands**
  - **cleartool setview *ViewName* - Launches ClearCase and displays the user's view.**
  - **cd *PathName* - Changes directory to a subdirectory in the VOB.**
  - **cleartool checkout -nc . - Check out a directory from ClearCase**
  - **cleartool checkout -nc *FileName* - Check out a file from ClearCase.**
  - **cleartool uncheckout - Cancels a checkout.**

Checking Out an Element from ClearCase

Access ClearCase by typing:
cleartool setview *ViewName*

↓

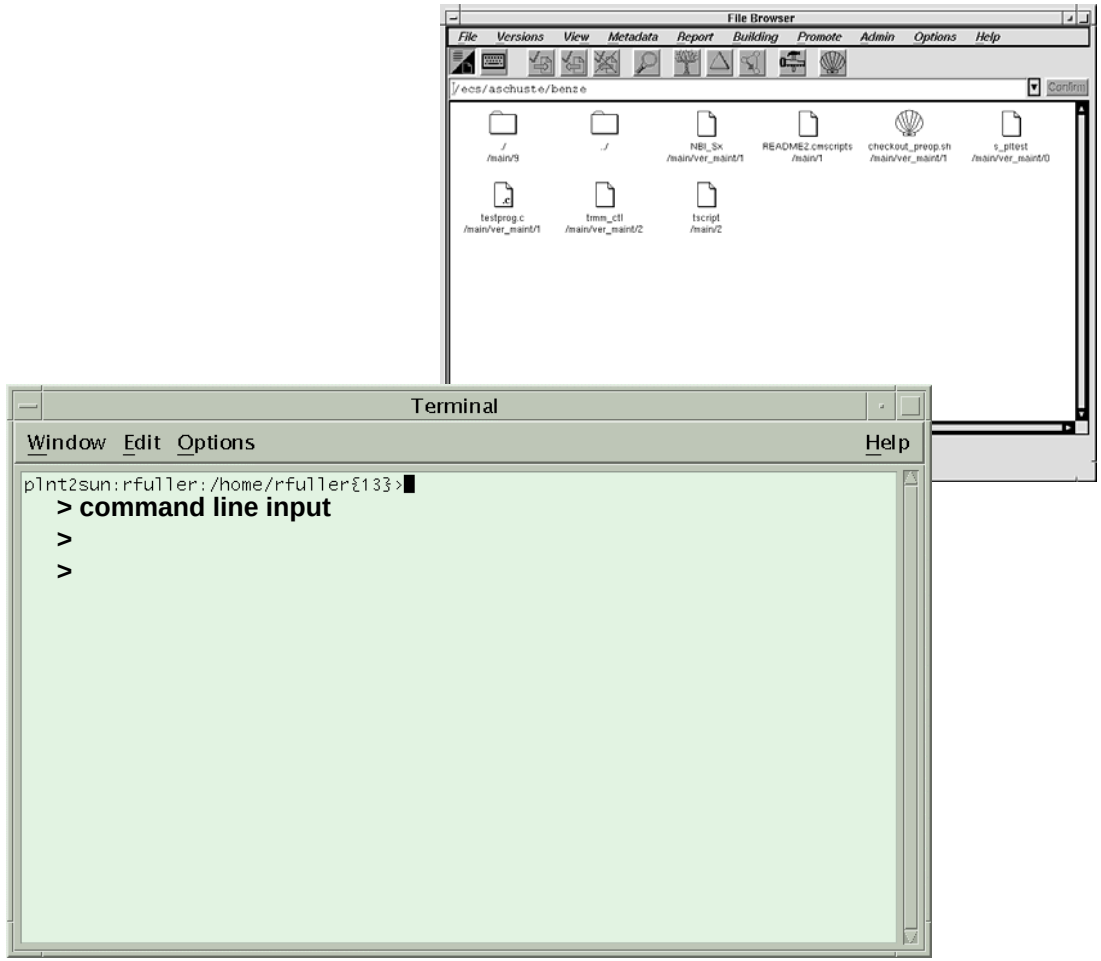
Change directories in VOB by typing:
cd *Pathname*

↓

Checkout directory by typing:
cleartool checkout -nc .

↓

Checkout file by typing: **cleartool checkout -nc *FileName***



Entering a Modified Element into ClearCase

- **Key Assumptions**
  - **A VOB and subdirectory has been created to hold the file.**
  - **A view has been created.**
  - **A file has been checked out from ClearCase and modified.**
- **ClearCase Commands**
  - **cleartool setview *ViewName* - Launches ClearCase and displays the user's view.**
  - **cd *PathName* - Changes directory to a subdirectory in the VOB.**
  - **cleartool checkin -nc *FileName* - Checks a modified file into ClearCase.**
  - **NOTE: DAAC policy may require a comment on entry of modified element into ClearCase.**

SSI&T PROCESS FLOW

- **Most steps in the SSI&T process are inter-related and some steps may assume that another step has been completed**
- **The ordering of the steps is very important but it cannot, however be interpreted as a detailed, step-by-step guide to SSI&T activities**
- **Refer to the Science Office Project Instructions in the EMD Process Assets Library (PAL)**
- **The Process Assets Library is accessible on the web at the following URL:**
  - **http://dmserver.hitc.com/EMD_PAL/index.html**

Acquiring the Delivered Algorithm Package by FTP

Log into **sgi machine**

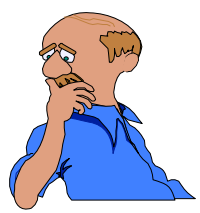
Type **cd *PathName***

Type ***ftp machine IPaddress***

Type ***user name*** and ***password***

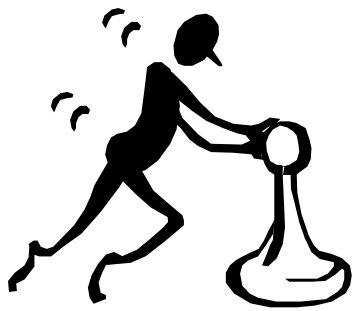
Type **cd *PathName*** where DAP is located

Type **binary** and **get *DAPFilename***



SSIT Team

Note: Tapes are also used to transfer files. Usually performed by the SA.



FTP Transfers files to the desired directory within the **sgi machine**

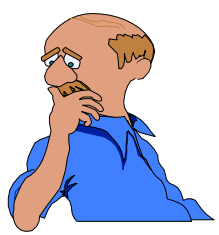
sgi machine designated for SSI & T

Unpacking the Delivered Algorithm Package

Log into **sgi machine**

Type **cd** ***UnpackPathName***

Type ***tar xvz PackedDAP***



SSIT Team

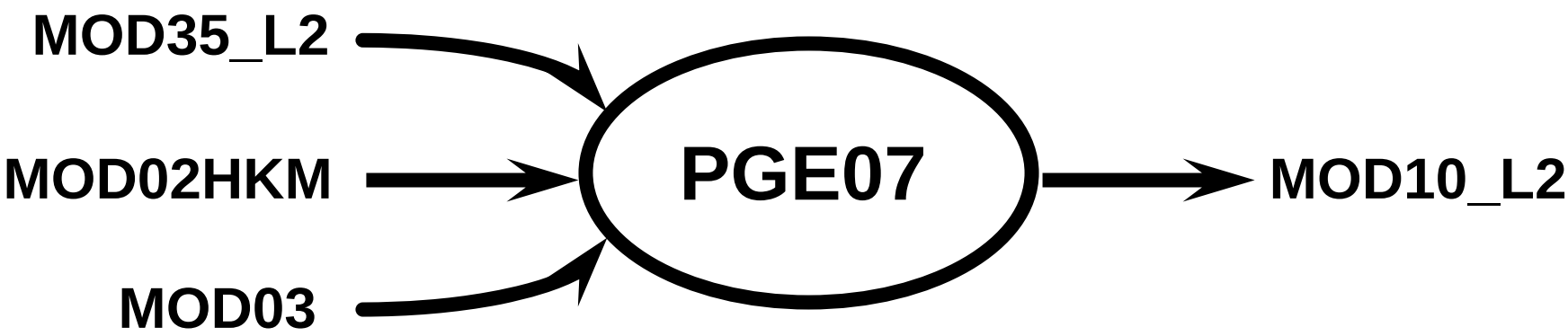
The tar file is unpacked in the desired directory within the **sgi machine.**

DESTINATION: INGEST

Delivered Algorithm Package

- **Typical Contents of DAP:**
  - **Source code**
  - **Message files**
  - **Make or build files**
  - **Shell Scripts**
  - **Process Control File (PCF)**
  - **Metadata Configuration File information to build an (MCF)**
  - **Instructions for building and running PGE**
  - **New metadata and ESDT Descriptor files**
  - **Test data for input and comparisons**

Training Example DAP
MODIS PGE07



All files: 5 min data

Training Example DAP (1 of 3)

Filename	Description
SDSRV:	
DsESDTMoMOD03.001.desc	Input ESDT descriptor file
DsESDTMoMOD02HKM.001.desc	Input ESDT descriptor file
DsESDTMoMOD35_L2.001.desc	Input ESDT descriptor file
DSESDTMoMOD10_L2.001.desc	Output ESDT descriptor file
libDsESDTMoMOD03.001.Sh.so	Shared library for input ESDT
libDsESDTMoMOD02HKM.001.Sh.so	Shared library for input ESDT
libDsESDTMoMOD35_L2.001.Sh.so	Shared library for input ESDT
libDsESDTMoMOD10_L2.001.Sh.so	Shared library for output ESDT

Training Example DAP (2 of 3)

Filename	Description
PDPS:	
PGE07.tar	PGE executable
PGE07.tar.met	Target MCF for PGE executable
PGE_PGE07#1.0#01.odl	ODL for PGE07
ESDT_ MOD03#2.0.odl	ODL file for binary input granule
ESDT_ MOD02HKM#2.0.odl	ODL file for binary input granule
ESDT_ MOD35#2.0.odl	ODL file for binary input granule
ESDT_ MOD10_L2#2.0.odl	ODL file for binary output granule
MOD02HKM.A1996218.1555.002.hdf	Binary input data granule
MOD03.A1996218.1555.002.hdf	Binary input data granule
MOD35_L2.A1996218.1555.002.hdf	Binary input data granule

Training Example DAP (3 of 3)

Filename	Description
To be generated at run time:	
MOD02HKM.A1996218.1555.002.hdf.met	Target MCF for binary input granule
MOD03.A1996218.1555.002.hdf.met	Target MCF for binary input granule
MOD35_L2.A1996218.1555.002.hdf.met	Target MCF for binary input granule
MOD10_L2.A1996218.1555.002.mcf	MCF for output product
MOD10_L2.A1996218.1555.002.mcf.met	Target MCF for status

SSIT Manager

- **Provides a common interface to the SSI&T software tools and manages their operation**
  - **Setup SSI&T Manager and checklist.**
  - **Open xterm session.**
  - **Code Analysis.**
  - **Office Automation Tools.**
  - **Standards Compliance.**
  - **Product Examination using EOSView and IDL.**
  - **File Comparison in HDF, binary or ASCII format.**
  - **Edit Text file.**
  - **Initialize and Update PDPS database.**
  - **Data Server Access.**

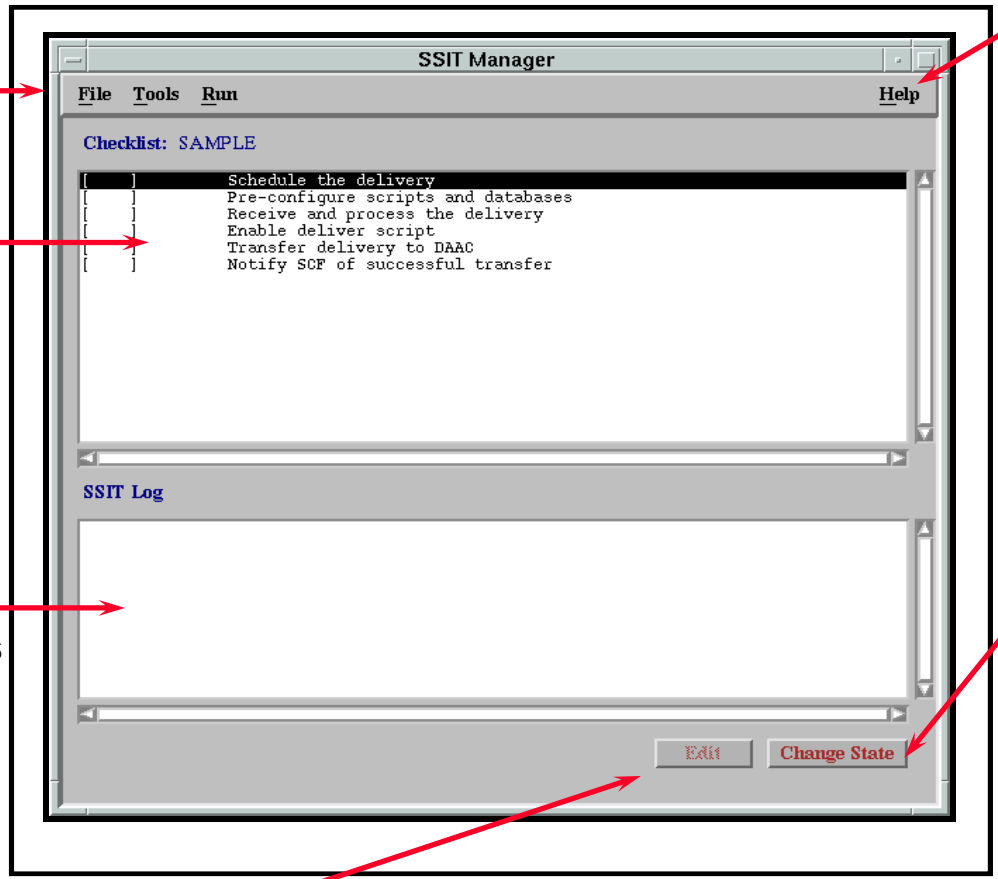
Setup of SSIT Manager

- **Configuration of Environment**
  - **SSIT Manager runs only on Sun platforms in the subsystem DPS**
  - **User makes a local copy of a Process Control File (PCF) for SSIT Manager and repeat same for a sample checklist.**

Setup of Checklist for SSIT Manager

- **Steps to Setup the SSIT Manager Checklist for Use in SSI&T**
- **From the SSIT Manager, select Tools → Product Examination → EOSView**
  - **The EOSView GUI will be displayed**

SSIT Manager GUI



Menu Bar: Allows access to SSI&T Tools

Checklist Pane: List set of steps to be completed

Log Pane: Log of activities accomplished

Help: Provides access to help features

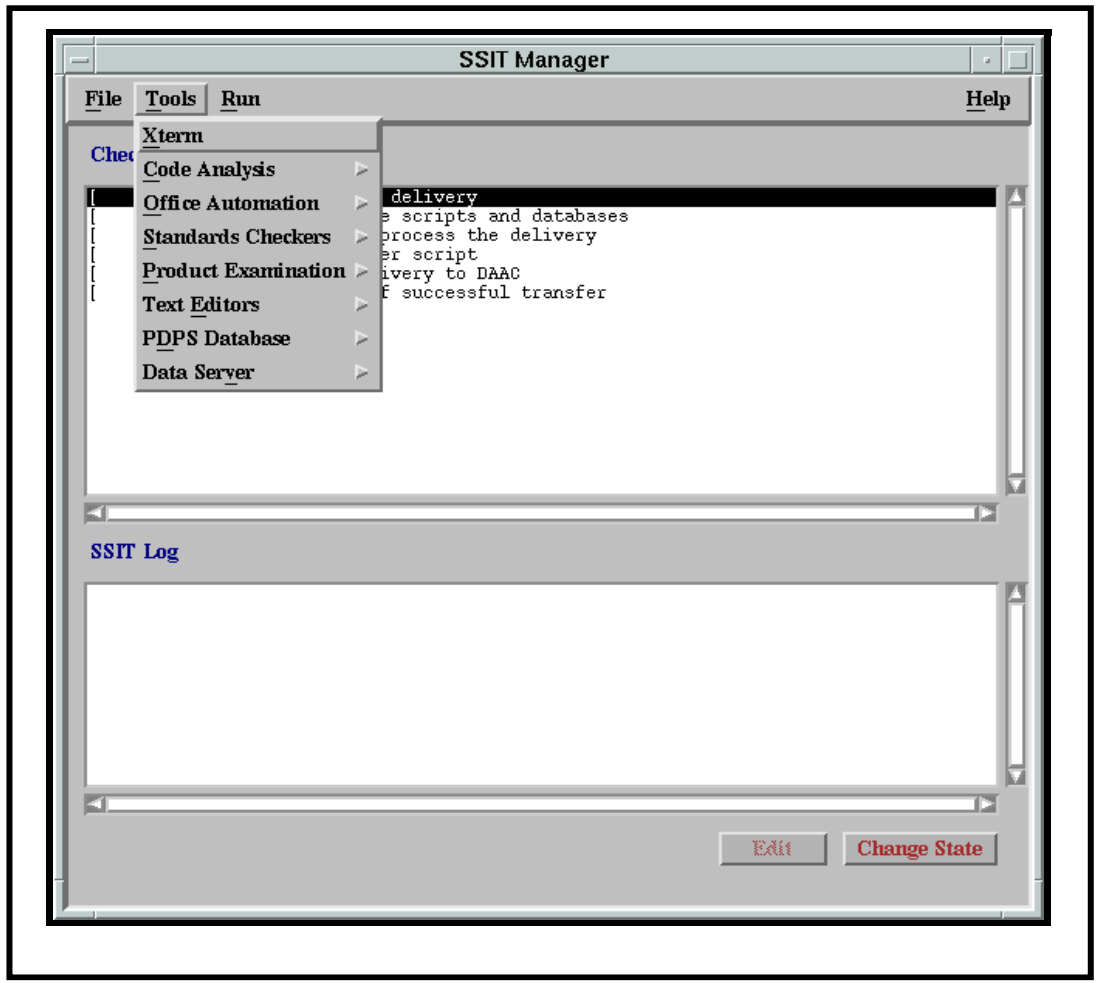
Change State: Button allows Checklist state to Toggle

Edit: Button allows Checklist to be edited

SSIT Manager Tools

Tools:

1. **Xterm:** Starts an Xterm window session
2. **Code Analysis:** Performs static code analysis
3. **Office Automation:** MS Windows, MS Office, MS Excel
4. **Standards Checkers:** Prohibited Function & Process Control File Checkers, ForCheck, Prolog Extractor
5. **Product Examination:** File Comparison Tools and EOSView
6. **Text Editors:** Emacs or Xedit Tools
7. **PDPS Database:** PCF ODL Template Tool, Science Metadata & Opnl Metadata Update
8. **Data Server:** Register Subscription, Insert Static, Insert Test Dynamic, Insert EXE TAR



Standards Checking

- **Purpose: Verify that the source files of a PGE are compliant with the ESDIS Data Production SCF Standards and Guidelines.**
- **Key Terms:**
  - **SDP Toolkit**
    - **provides an interface to the production system**
    - **allows science software to be portable to different platforms**
    - **reduces redundant coding at the SCF**
    - **provides value added functionality for science software development**

Standards Checking (Cont.)

- **Mandatory SDP Toolkit Functions**
  - **Error and Status Message Facility (SMF).**
  - **Process control Tools.**
  - **Generic Input/Output Tools.**
  - **Memory Allocation Tools.**
- **Optional SDP Toolkit Functions**
  - **Ancillary Data Access.**
  - **Celestial Body Position Coordinate System Conversion.**
  - **Constant and Unit Conversion.**
  - **IMSL.**

Standards Checking (Cont.)

- **Steps for Standards Compliance**
  - **FORTRAN 77 - On the AIT Sun.**
    - **Source FORCHECK setup file.**
    - **Create FORCHECK run script.**
    - **Invoke FORCHECK run script.**
    - **Examine the list file.**
  - **Fortran 90 and C - On the SDPS SGI.**
    - **Set environment to appropriate SDP Toolkit.**
    - **Compile the PGE using compiler flags.**
    - **Examine the list file.**
  - **Ada**
    - **Compile using COTS Verdix Ada Development System or GNU C Compiler, gcc.**

Prohibited Function Checker

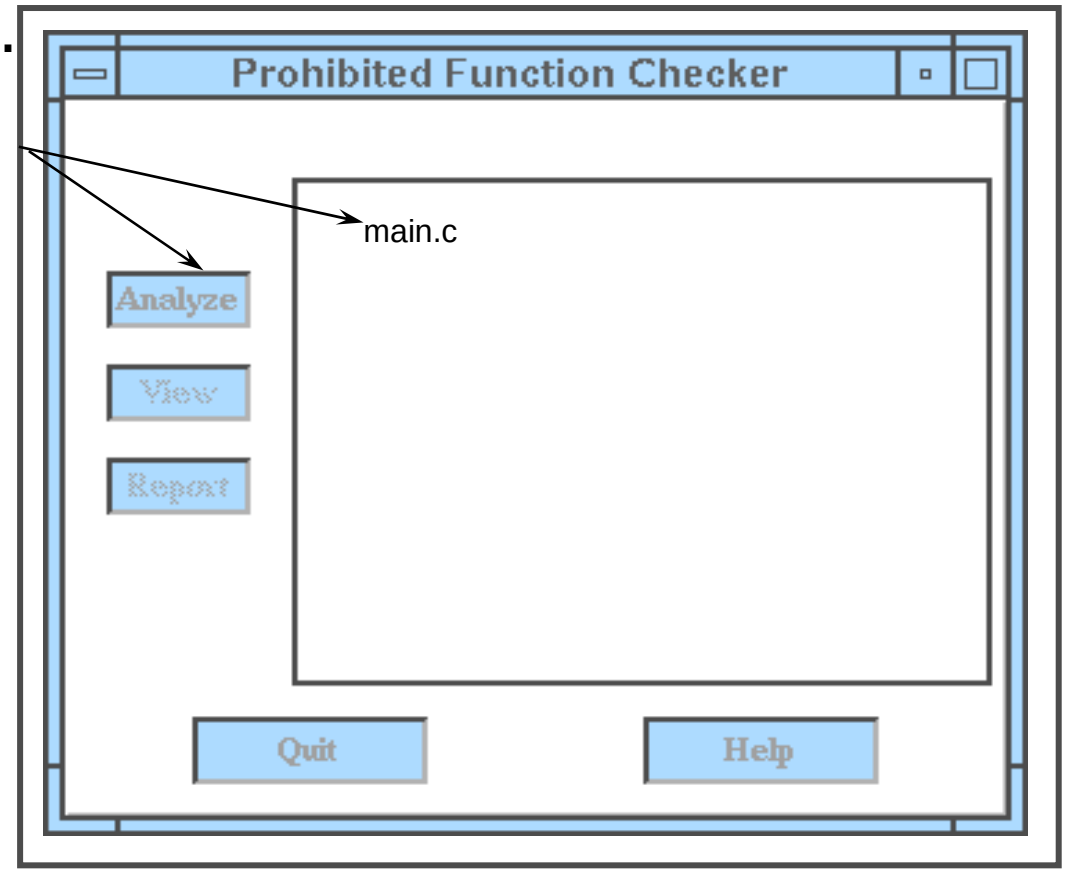
- **Used to check source files for the occurrence of functions that are prohibited in the DAAC production environment**
- **Key Procedure Commands**
  - **SSIT Manager**
    - **Tools → Standards Checkers → Prohibited Function Checker**
- **Run Analyze from GUI**
  - **Highlight files to be analyzed**
  - **Run checker**
  - **Generate report**
  - **Save and examine report**

Prohibited Function Checker GUI

Highlight files to be analyzed.

Run checker.

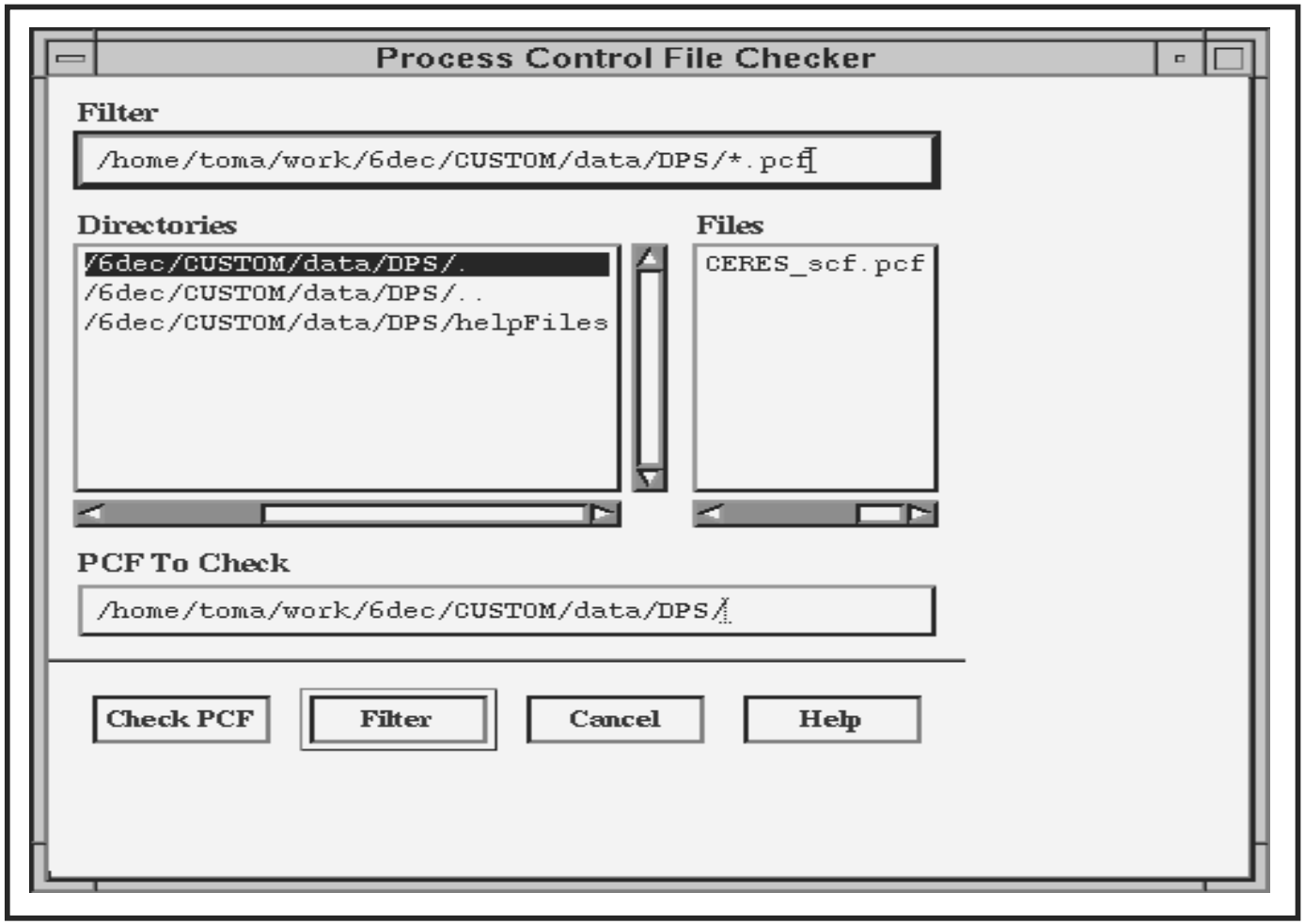
Generate report.



Checking Process Control Files

- **Key Procedure Commands**
  - **SSIT Manager**
    - **Tools → Standards Checker → Process Control File Checker**
- **Run the PCF Checker GUI**
  - **Select the directory**
- **The PCFs must be checked to verify that they are syntactically correct and contain all the information for the PGEs to run within the DAAC production environment**
- **Select one PCF and select the Check PCF button**
  - **Save or print the results file and examine results**

Checking Process Control Files



Extracting Prologs

- **The Prolog Extractor will search recursively for files with valid filename extensions. The beginning and end delimiters are:**
  - **!F77 !F90 !C !C++ !F77-INC !F90-INC !C-INC !PROLOG**
  - **!END**
- **Key Procedure Commands:**
  - **SSIT Manager**
    - **Tools → Standards Checker → Prolog Extractor**
- **Run the Prolog Extractor GUI.**
  - **Select the directory with source files.**
  - **Save or print the output Prologs files.**

Compiling and Linking Science Software

- **Science software developed at SCFs using the SDP Toolkit needs to be compiled and linked first with SCF Toolkit version to compare results at each facility**
- **Then the science software needs to be compiled and linked with the DAAC Toolkit**

Compiling and Linking Science Software (Cont.)

- **Preparation for compile and link:**
  - **Source correct SDP Toolkit library version - total of 8 versions.**
    - **Location Type: SCF or DAAC**
    - **Computer Language Type: FORTRAN 77, Fortran 90, C, Ada**
    - **Object Type: 32-bit mode or 64-bit mode, Old 32 bit (SCFs only)**
  - **New and old 32-bit modes are distinguished by compiler options.**
  - **Update PCF for execution of PGEs at the DAAC.**
  - **Compile Status Message Files.**

Updating a PCF

- **PCF sections:**
  - **System Runtime Parameters**
  - **Product Input**
  - **Product Output**
  - **Support Input**
  - **Support Output**
  - **User-defined Runtime Parameters**
  - **Intermediate Input**
  - **Intermediate Output**
  - **Temporary I/O**
- **Update appropriate path names where necessary:**
  - **Add 10111|ShmMem|~/runtime||||1**

Updating a PCF

Checkout directory and file by typing:
cleartool checkout -nc .
cleartool checkout -nc *PCFFileName*

↓

Enter the vi editor by typing:
vi *PCFFileName*

↓

Edit the PCF.

↓

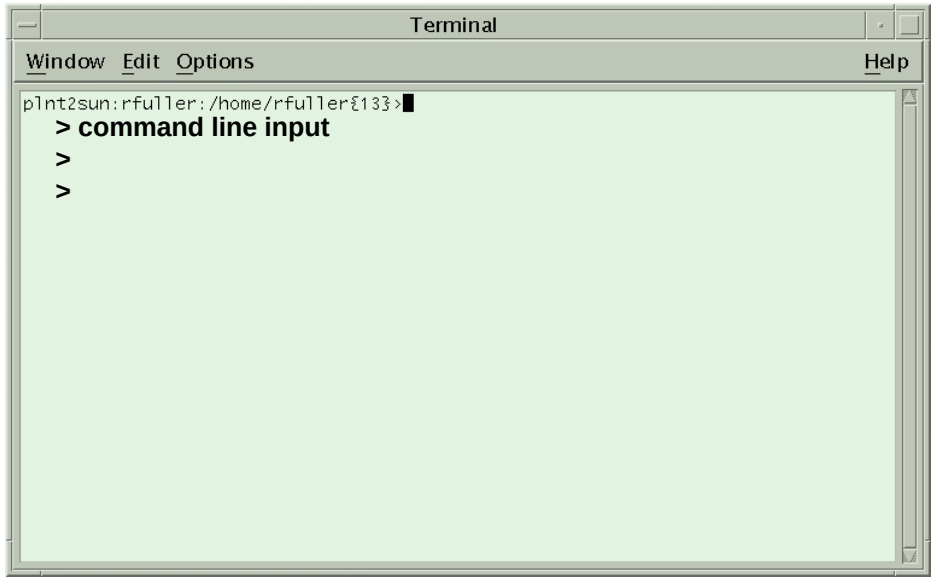
After editing, re-run the PCF Checker by selecting **Tools→Standards Checker→ PCF Checker** and from the SSIT menu.

↓

Checkin the file by typing:
cleartool checkin -nc *PCFFileName*

→

Checkin the directory by typing:
cleartool checkin -nc .



Compiling the SMF

- **Status Message Facility (SMF) Files - Also known as Error Status Message.**
- **Provides:**
  - **An error and status message handling mechanism**
  - **A method to send log files, informational messages and output data files to DAAC personnel or remote users.**
- **SMF files need to be compiled with science software into message files and include files.**
- **These files will be used by science software during runtime.**
  - **smfcompile -*lang* -f *TextFile*.t**
  - **-*lang* is the computer language and *TextFile*.t is the SMF file.**

Compiling the SMF (Cont.)

- **Process Steps:**
  - **Set ClearCase view (if source code is in ClearCase).**
  - **Set up SDP Toolkit environment.**
  - **Go to SMF directory for the PGE.**
  - **Run the SMF compiler.**
  - **Move created files to proper directories.**

Compiling a PGE and Linking with SCF and DAAC SDP Toolkits

- **Compiling and Linking of Science Software will vary according to the instructions from the Instrument Software Development Teams.**
- **Compiling and linking with SCF and DAAC versions differs only in the setting of the SDP Toolkit environment.**

Compiling a PGE and Linking with SCF and DAAC SDP Toolkits (Cont.)

- **Process Steps:**
  - **Read all instructional information supplied with the delivery.**
  - **Log into the SDPS SGI and set up the proper SDP Toolkit environment.**
  - **Set the ClearCase view if software is already in ClearCase.**
  - **Compile Status Message Facility files first.**
  - **Examine the make or build file and alter if necessary.**
  - **Using the make or build file, perform the build.**
  - **If make file has been changed, check in modified version.**

Running a PGE in a Simulated SCF Environment

- **Running a PGE at the DAAC in a simulated SCF environment should produce identical results as those at the SCF.**
- **Process Steps:**
  - **For SSI&T set up the SCF SDP Toolkit environment.**
  - **For SSI&T Training set up the DAAC Toolkit environment (results will be the same).**
  - **Set the environment variable PGS_PC_INFO_FILE to path and file name of PCF for the PGE.**
  - **If the PGE has been run before in the same directory, remove old log files.**
  - **Run the PGE from the command line.**

Running a PGE in a Simulated SCF Environment (Cont.)

- **To capture PGE runtime statistics for the PDPS Database, perform profiling using the DpPrRusage Program (Rusage).**
- **Statistics needed:**
  - **wall clock time**
  - **user time**
  - **system time**
  - **amount of memory used**
  - **number of page faults**
  - **number of input and output blocks**
  - **number of swaps**

Available SDP Toolkits Used by the SGI Science Processors

SDP Version	Language Type	Library Object Type	$PGSBIN
SCF	C++ or C	Old 32-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi_scf_cpp
SCF	FORTRAN 77 or C	Old 32-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi_scf_f77
SCF	Fortran 90 or C	Old 32-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi_scf_f90
SCF	Thread	Old 32-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi_scf_r
SCF	C++ or C	New 32-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi32_scf_cpp
SCF	FORTRAN 77 or C	New 32-bit Mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi32_scf_f77
SCF	Fortran 90 or C	New 32-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi32_scf_f90
SCF	Thread	New 32-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi32__scf_r
SCF	C++ or C	64-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi64_scf_cpp
SCF	FORTRAN 77 or C	64-bit Mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi64_scf_f77
SCF	Fortran 90 or C	64-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi64_scf_f90

Available SDP Toolkits Used by the SGI Science Processors (Cont.)

SCF	Thread	64-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi64_scf_r
DAAC	C++ or C	Old 32-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi_daac_cpp
DAAC	FORTRAN 77 or C	Old 32-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi_daac_f77
DAAC	Fortran 90 or C	Old 32-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi_daac_f90
DAAC	Thread	Old 32-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi_daac_r
DAAC	C++ or C	New 32-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi32_daac_cpp
DAAC	Fortran 77 or C	New 32-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi32_daac_f77
DAAC	Fortran 90 or C	New 32-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi32_daac_f90
DAAC	Thread	New 32-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi32_daac_r
DAAC	C++ or C	64-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi64_daac_cpp
DAAC	Fortran 77 or C	64-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi64_daac_f77
DAAC	Fortran 90 or C	64-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi64_daac_f90
DAAC	Thread	64-bit mode	$CUSTOM_HOME/TOOLKIT/toolkit/bin/sgi64_daac_r

Running a PGE in a Simulated SCF Environment (Cont.)

Sample of an Rusage File produced:

P0spg01{emcleod}6: more Profile.out

source .cshrc

 # cd TEST/MOD*

 # ls

#

/usr/ecs/OPS/CUSTOM/bin/DPS/EcDpPrRusage MOD_PR10.exe > Profile.out

p0spg01{emcleod}9: more profile.out

 # Resource Usage Information

COMMAND=MOD_PR10.exe

EXIT_STATUS=0

ELAPSED_TIME=233.583145

USER_TIME=10.046158

SYSTEM_TIME=7.555547

MAXIMUM_RESIDENT_SET_SIZE=4080

AVERAGE_SHARED_TEXT_SIZE=0

AVERAGE_UNSHARED_DATA_SIZE=0

AVERAGE_UNSHARED_STACK_SIZE=0

? **PAGE_RECLAIMS=151**

? **PAGE_FAULTS=0**

? **SWAPS=0**

? **BLOCK_INPUT_OPERATIONS=2**

? **BLOCK_OUTPUT_OPERATIONS=2710**

? **MESSAGES_SENT=0**

? **MESSAGES_RECEIVED=0**

? **SIGNALS_RECEIVED=0**

? **VOLUNTARY_CONTEXT_SWITCHES=1095**

? **INVOLUNTARY_CONTEXT_SWITCHES=2**

? **P0spg01{emcleod}10:**

Examining PGE Produced Log Files

- **PGEs produce three log files during runtime:**
  - **Status Log - captures all error and status information.**
  - **User Log - captures a subset of more informational messages.**
  - **Report Log - captures arbitrary message strings.**
- **Log file messages are written by both SDP Toolkit and science software using the Status Message Facility (SMF).**
- **Process Steps:**
  - **Examine PCF to get location of log files.**
  - **With SCF version of Toolkit, location and filenames can be set as desired.**
  - **Look for errors or warnings, anomalous messages**

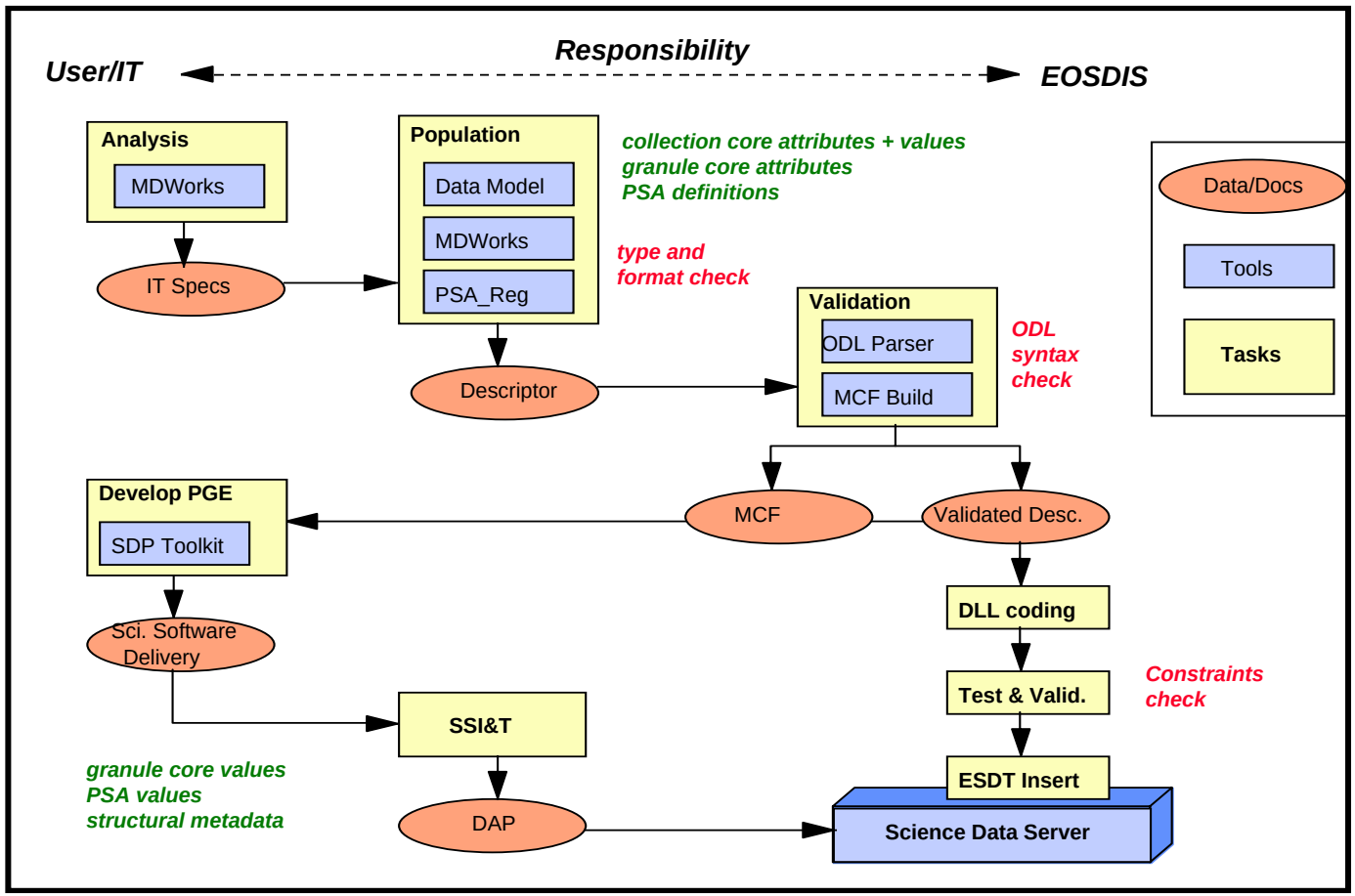
File Comparison and Data Visualization

- **An important activity for SSI&T is comparing the output data products from the PGE runs to test files delivered with the PGE.**
- **The comparison may consist of display of metadata in HDF files, display of differences in data values, or display of images of the data products.**
- **Searches are performed for any differences beyond specified tolerances.**
- **Data product files can be compared by a variety of tools accessible by the SSIT Manager GUI.**
  - **Tools → Product Examination → File Comparison → HDF [or ASCII or Binary]**

File Comparison and Data Visualization (Cont.)

- **Data visualization tools are accessible by the SSIT Manager GUI.**
  - **Tools → Product Examination → IDL or EOSView**

The Metadata Workflow



ESDT Management

- **In order for science data to be handled by the system, it must be formally described**
- **At the collection level, that description is the Earth Science Data Type or ESDT**
  - **Basically, when an ESDT is defined/installed to the data server subsystem, the SDSRV parses the descriptor into various portions needed by its own CSCIs, and other subsystems**
  - **An entry is made in the SDSRV database containing the meaning of the ESDT, each of its attributes, and each of its services (references to the executable DLLs)**
  - **The (Sybase) database managed by the SDSRV has sufficient information to satisfy queries sent to the SDSRV**

ESDT Management (Cont.)

- **The ESDT Descriptor file text contains the information mentioned above in an ODL format**
  - **The bulk of these files are placed in a given mode during the install process for that mode**
  - **They generally reside in directory /usr/ecs/*MODE*/CUSTOM/data/ESS**
  - **In order for these descriptors to be of any use, their information needs to be extracted and parsed to various subsystem databases (This is called the ESDT Install Process)**
- **ESDT Descriptor files may contain errors or the basic ESDT information is evolved such that the old descriptor information may have to be replaced or updated in the relevant databases**

Earth Science Data Types- ESDTs

- **Representation of different types of data products from the scientists perspective.**
- **Define to the Version 2 Data Model:**
  - **Collection level metadata attributes and values.**
  - **Granule level metadata attributes.**
  - **Data services appropriate to the ESDT collection and data granules within the collection.**

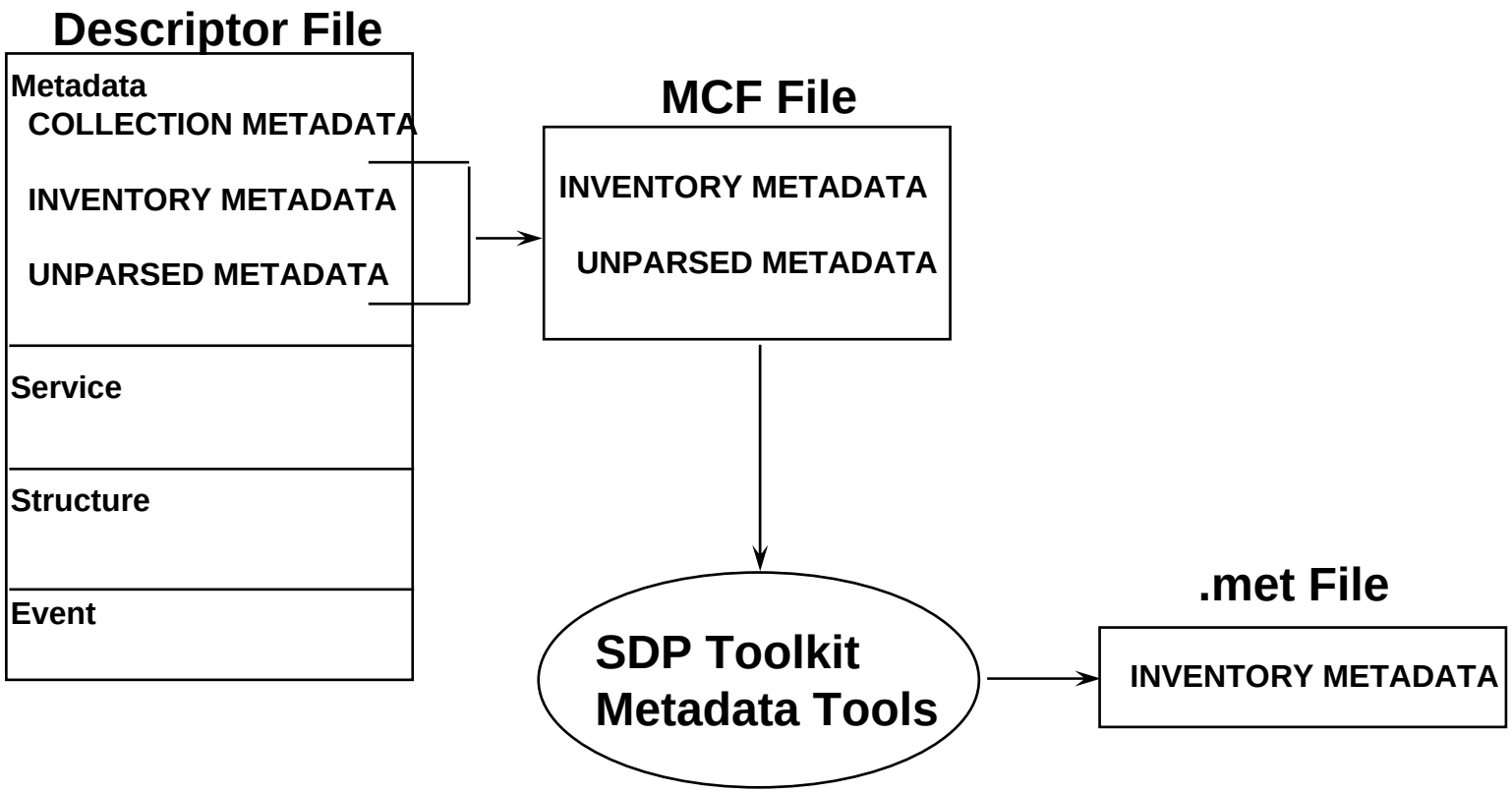
ESDT Components

- **Descriptor File:**
  - **Collection level metadata attributes and values.**
  - **Granule metadata attributes.**
  - **Granule metadata attributes valid values.**
  - **Services to be performed for the science data.**
  - **The set of attributes in the granule level part of the descriptor is the source for producing a Metadata Configuration File (MCF). From the MCF will be produced a .met file using the SDSRV.**
- **DLL File:**
  - **The Dynamic Link Libraries (DLL) file is uniquely produced for each ESDT and must be installed with each ESDT**

MCF Generation from SDSRV

- **The Metadata Configuration File (MCF) is produced by the following processes:**
  - **The ESDTs and DLL's are installed into the SDSRV with error checking taking place in the descriptor before installation**
  - **The GETMCF tool is executed to pull the MCF from the SDSRV**
  - **The actual MCF is generated and then copied from the Inventory and Archive section of the Collection Descriptor**

Descriptor, MCF & .met Files



What are Valids?

- Valids are type of integrity constraint to ensure that metadata values comply with the data model and database schema requirements
- Valids are used to ensure the data products quality and consistency of search queries
- Currently implemented for
  - Data Type and Length checking; e.g. STRING, FLOAT, etc.
  - Match Rules; e.g. DayNightFlag = (Day, Night, Both) ex: DAY will not match, resulting in a failed insert. (match rule is exacting)
  - Range Checking ; e.g.
    - Longitude Minimum = -180.00 to
    - Longitude Maximum = +180.00
  - Expressions; e.g. MinimumAltitude >= 0.0

Attribute Valid Processing

- **Attribute Valids have dependency with corresponding Descriptors, PGEs; Versions must be "in sync"**
  - **Attribute Valids values may have a dependency upon SDSRV version and potentially other subsystems (Client & Ingest for example)**
  - **Changes to Valids may impact code; Versions must be "in sync"**
  - **Attribute Valids are stored within SDSRV metadata database**
- **Incoming metadata validation processing is imbedded within the SDSRV software**
  - **Action on ESDT/granule with invalid metadata depends on "MANDATORY=" setting**

Attribute Valid Processing (Cont.)

- **Currently, incoming metadata checked against one of the four basic constraints checks as specified**
  - **Match**
  - **Range**
  - **Expression**
  - **NONE)**
    - **“NONE” means no value checking is performed**
- **Messages are logged for attributes containing invalid data. Action on ESDT/granule with invalid metadata depends on “MANDATORY=“ setting**

PSA Process

- **Product Specific Attributes (PSA) information is obtained from Instrument Teams through a template (PSA Template) that has been provided to them**
- **Data Engineering performs analysis to verify that the PSAs are unique and conform to the Data Model**
- **Approved PSAs are submitted to the EMD CCB for approval to update the PSA baseline**
- **PSA_Registry database is updated with the new approved PSAs**
- **Reports are generated on a bi-weekly basis or on as-needed basis**

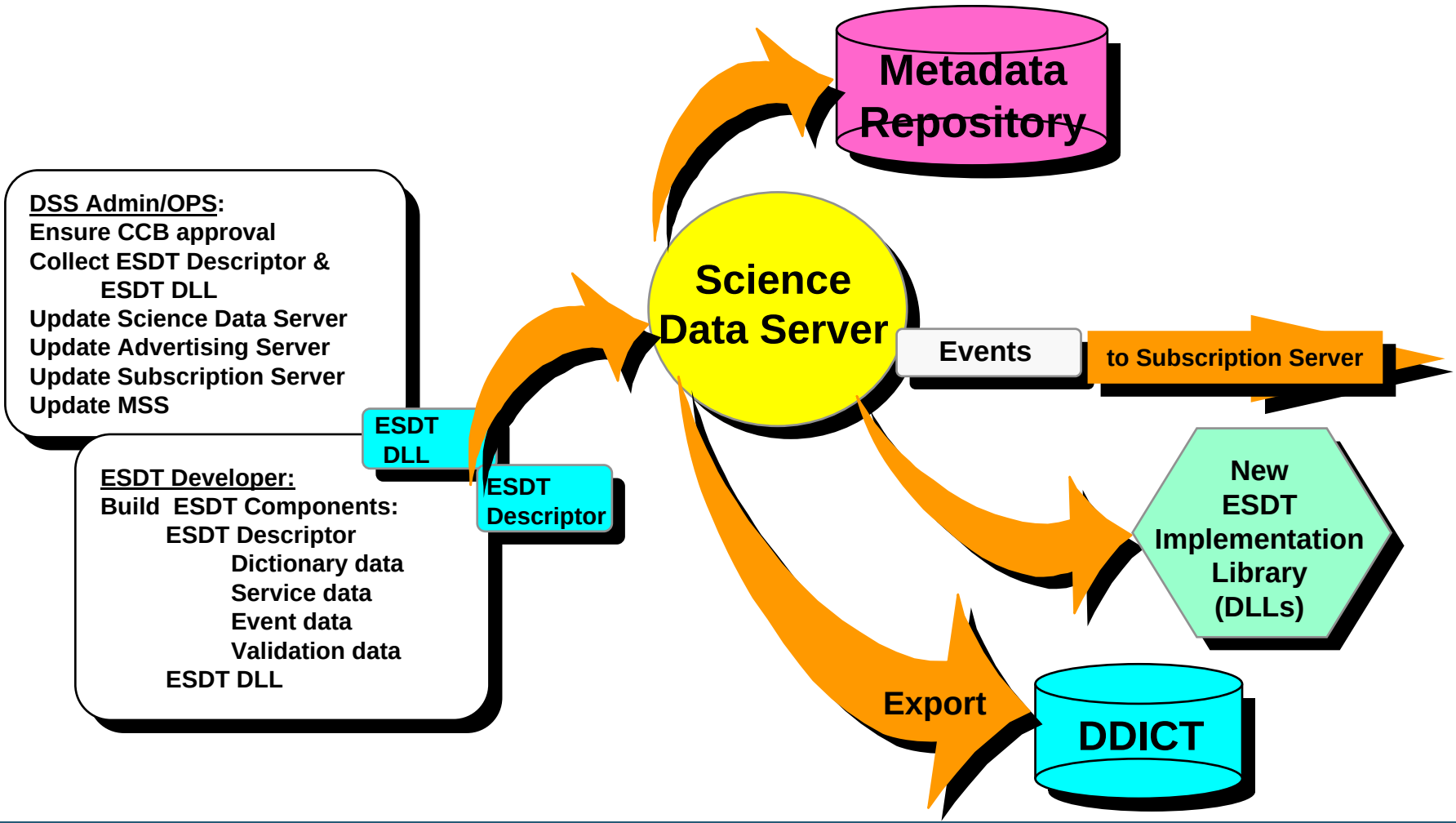
ESDT CM Process

- **Stored in ClearCase**
- **Directory structure is established to differentiate the different versions of ESDTs**
  - **Allows support of multiple versions of Database valids and schema**
  - **Allows for the creation of custom ESDT changes in order to support the evolution of code development**
  - **Makes delivery to different sites/platforms (PVC, VATC, GSFC, etc…) easier**
- **Changes to ESDTs are based upon input from development, and ITs**
- **Modified ESDTs are merged onto the baseline after approval at the merge meetings**

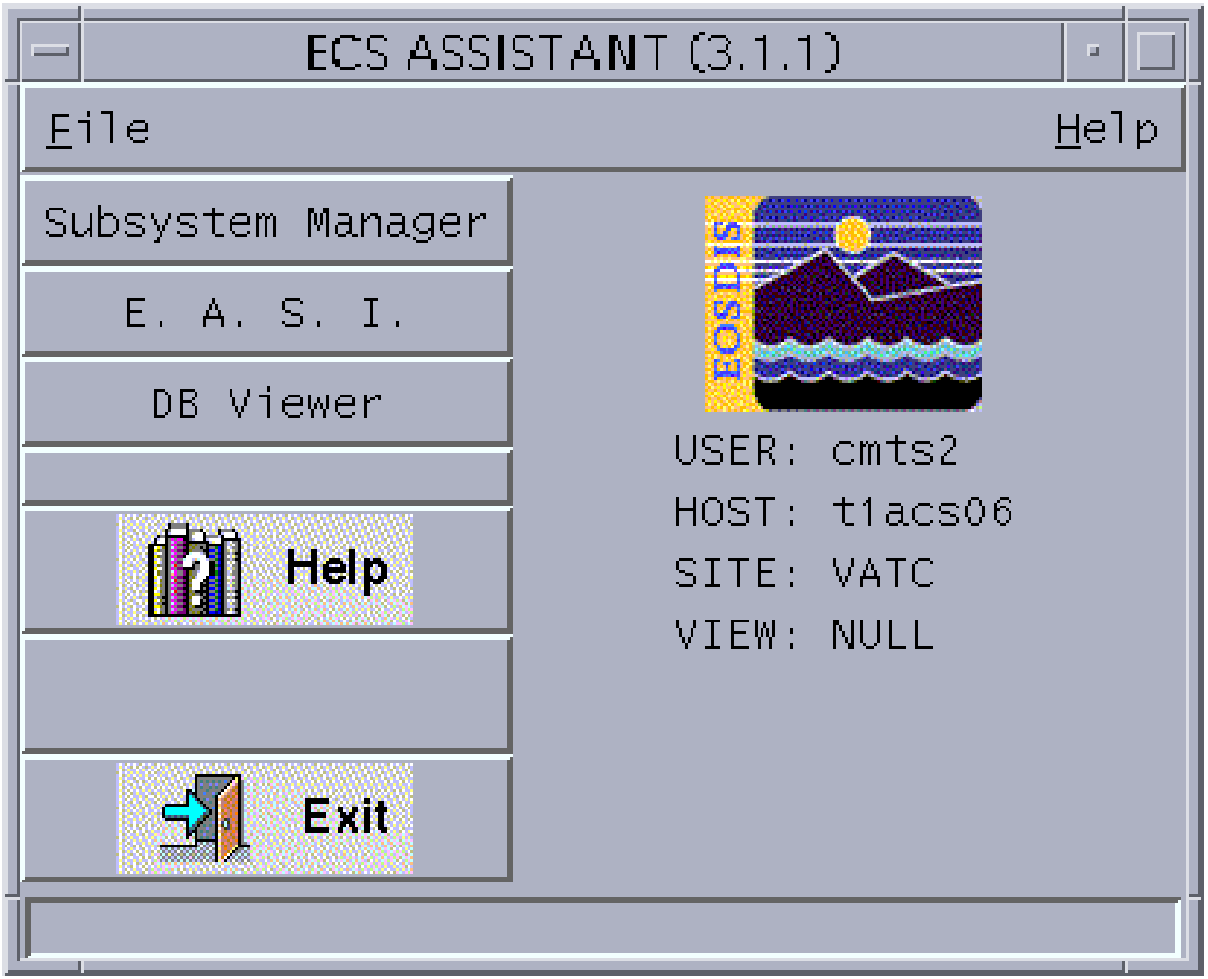
Overview: ESDT Development and Installation Process

- **Complete ESDT development and installation process involves the following steps:**
  - **ESDT Generation**
  - **ESDT Installation in the SDSRV**
    - **Both the Descriptor and DLL are stored within the Science Data Server**
    - **From the Science Data Server, the attribute information contained in the ESDT Descriptor is passed into a number of Clients as depicted on the next slide (Adding a new ESDT)**

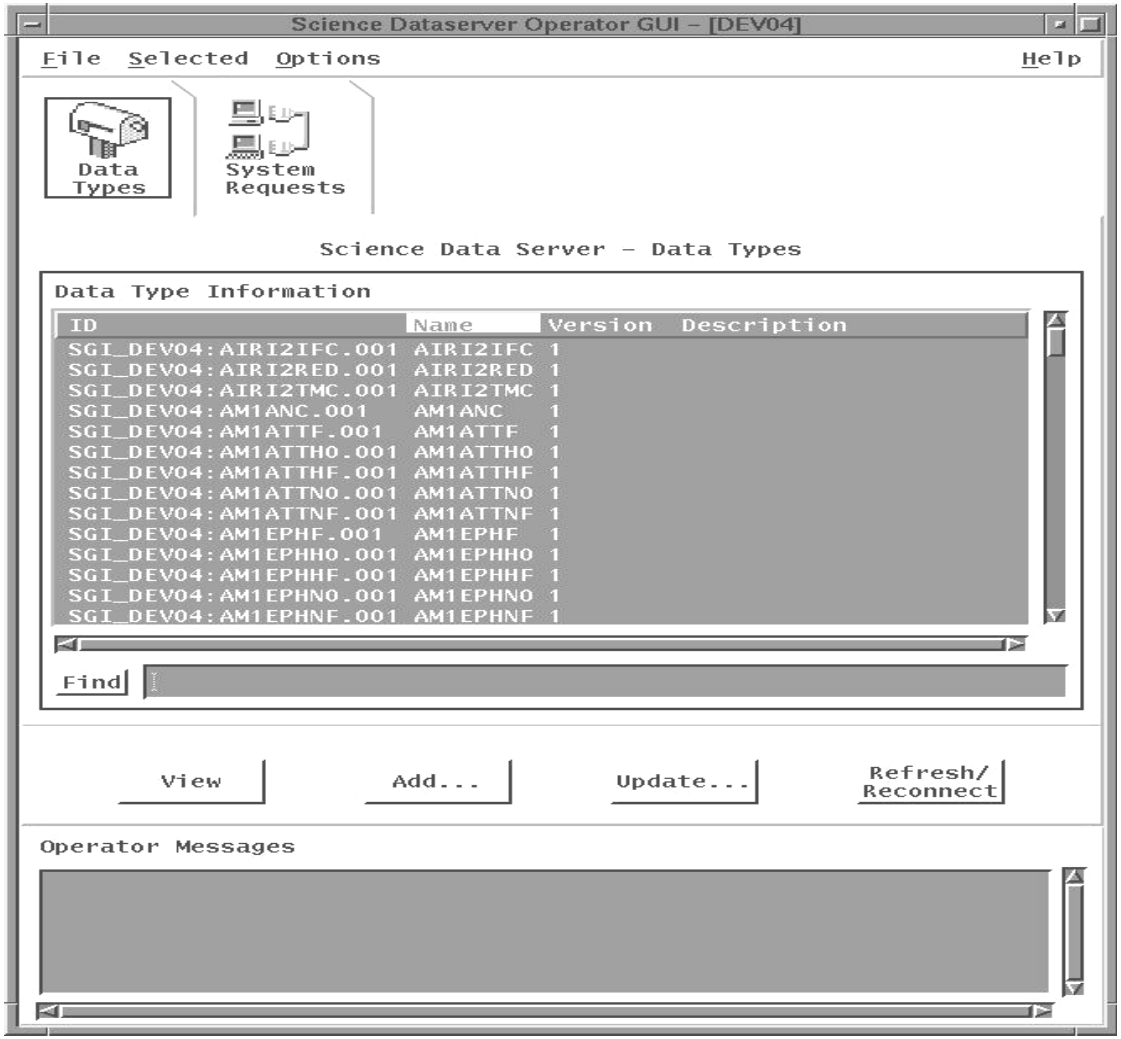
Adding a new ESDT: Operational Overview



ECS ASSISTANT



Installing an ESDT/DLL Using Science Data Server Operator GUI



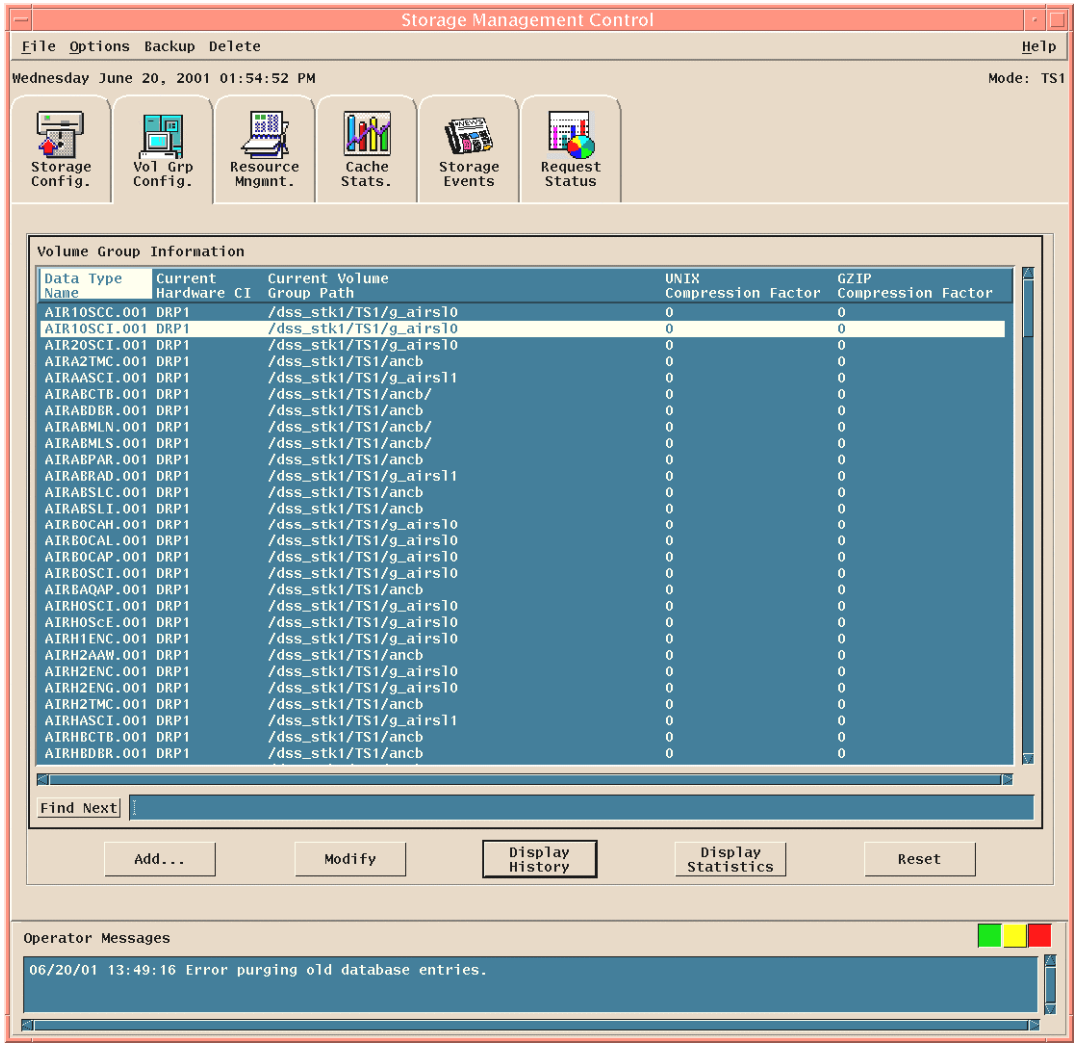
Viewing and Copying ESDT/DLL using ECS Assistant GUI

- **Key Assumptions**
  - **The ECS Assistant is up with the necessary servers listening.**
  - **SSI&T personnel have permissions and privileges to register ESDTs.**
  - **The ECS Assistant GUI is running with ESDT Manager selected.**
  - **The ESDT and DLL descriptor files are installed in the specific mode.**
  - **ESDT Descriptors Located: /usr/ecs/TS1/CUSTOM/data/ESS**
  - **DLL located: /usr/ecs/TS1/CUSTOM/lib/ESS**
  - **The ESDTs and universal DLL descriptor file are normally installed dynamically as one when the ESDT is registered using the Science Data Server Operator GUI.**

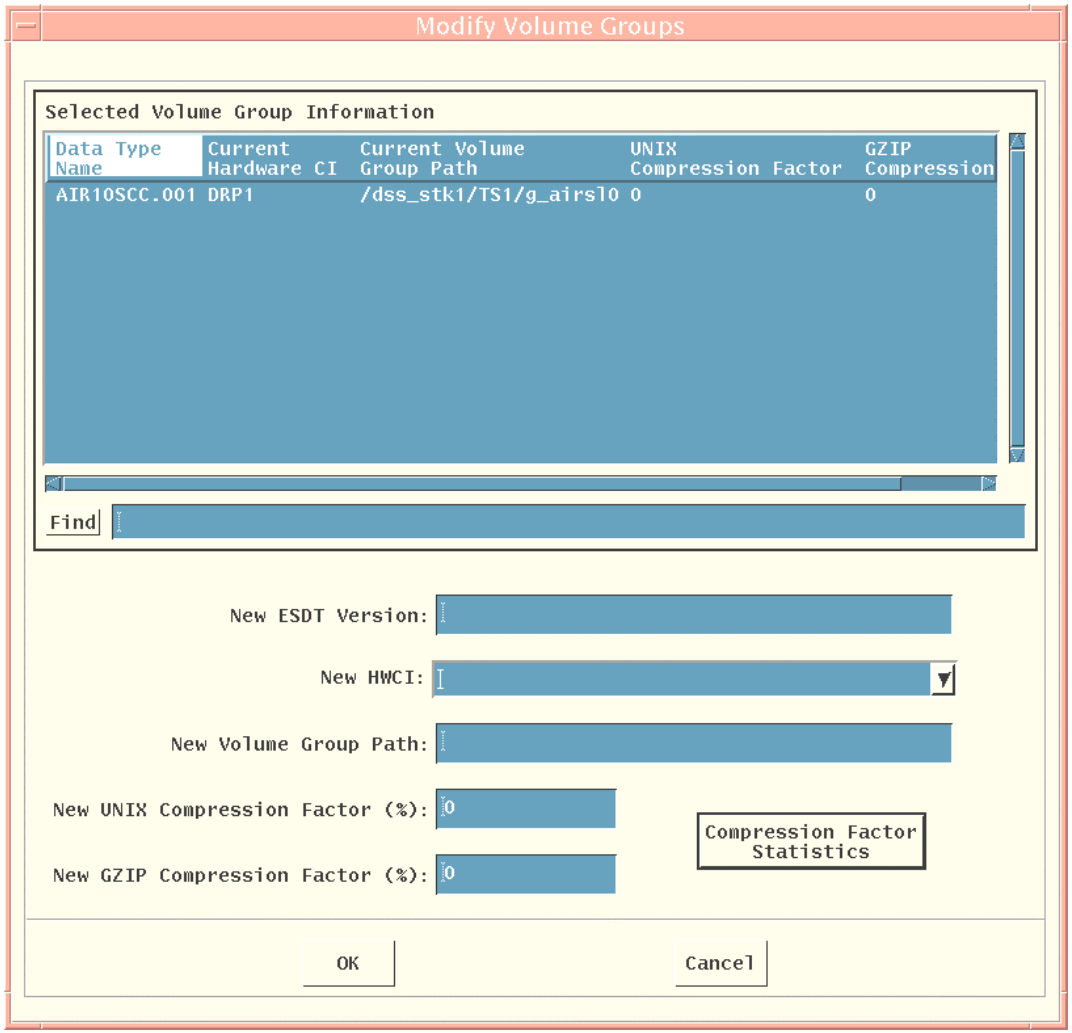
ESDT Volume Group Configuration

- **Once an ESDT is installed, the system knows how to deal with the collections and granules associated with that ESDT - up to a point**
  - **The Storage Management subsystem needs some additional information for its database so that it knows where to archive and retrieve the data associated with a given ESDT**
    - **This is the ESDT Volume Group information**
    - **When an insert or acquire is performed, Storage Management needs to know which HWCI (Hardware CI) and directory are involved**
  - **The Volume Group information can be created and modified using the Storage Management GUI**

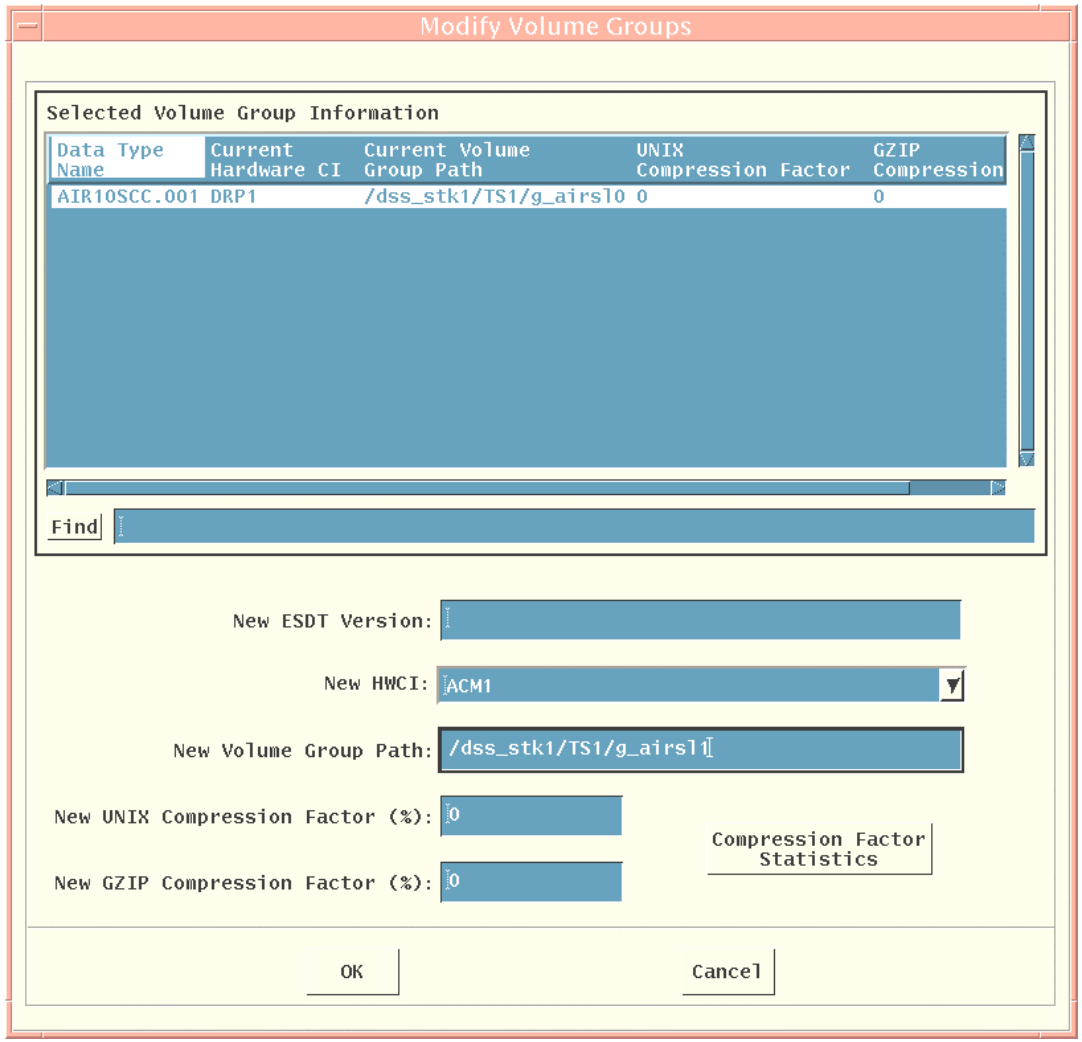
Storage Management GUI



Modify Volume Groups GUI



Modify Volume Groups GUI with Entries



Production Rules

- **Production rules are the instructions about how a particular PGE is to be run**
  - **The instructions specify a wide range of information such as the input and output data types, the frequency of execution, activation conditions, and error handling instructions**
- **Production rules are described in detail in 625-EMD-006, Training Material for the EMD Project Volume 6: Production Planning and Processing**

DPREP

- **Data Preprocessing (DPREP) is a vital part of SSI&T**
- **A general description of DPREP for Terra, Aqua, and Aura is provided in 625-EMD-006, Training Material for the EMD Project Volume 6: Production Planning and Processing, and has not been duplicated in this lesson**

Updating the Orbit Model

- **To determine real-time the latest Orbit Start times, Orbit Period, Path Number and Orbit Number, PDPS takes in specific information about the orbit of the satellite during initial SSI&T**
  - **This information then becomes the basis for predictions of future orbit start times and numbers**
  - **Because this value is accurate within a fraction of a second of time, the satellite may "drift" or a correction to orbit, known as a "burn" may have been applied**
  - **Therefore, the satellite Orbit Start Time can get out of sync either +/- with reality**
  - **The consequences are an elapse in time that will affect the Production Request Editor's ability to find a granule that should match with a DPR, or an incorrect Orbit Time could be passed to the PGE**

Updating the Orbit Model (Cont.)

- **The update of Orbit parameters will be done weekly at a specific time with scripts specifically written to extract the new Orbit Parameters from the most recent DPREP output file**
- **These parameters in turn will be inserted manually to the ORBIT.ODL file**
- **Then SSI&T personnel will re-register the Orbit.ODL file in PDPS**

Updating the Orbit Model (Cont.)

- **The DAAC Support Help Desk team is responsible for knowing when changes to Orbit location have taken place from the Flight Dynamics System (FDS)**
  - **A KnowledgeBase with backup procedures will be maintained for contingencies concerning Orbit Model updates**
- **DPREP processing will be the most likely place to experience a failure due to Orbit time sync error encounters**
  - **The restoration of Orbit parameters with new values from FDS will most likely be necessary**

PGE Registration

- **PGE Registration is the step during Science Software Integration and Test (SSIT) that defines a PGE to PDPS.**
- **Information about the PGE is put into ODL files for ingestion into PDPS.**
  - **These files are read by SSIT software.**
  - **The data describing the PGE is stored in the PDPS database.**

PGE Registration (Cont.)

- When the PGE is executed in the production environment, the information in the PDPS database is retrieved to schedule the PGE for execution.
  - The information tells PDPS when the PGE should be scheduled, what data needs to be present to run the PGE, and what processing resources are needed.
- The Instrument Teams may be asked to fill out a web page that describes their PGE so that the SSIT operators can properly create the ODL files.

PGE Registration ODL format

- **ODL is simply a parameter=value file format.**
  - **Each line in the file (except for comments) is of the form: parameter = “value”**
- **There are currently five types of ODL files that need to be specified for PGE registration.**
  - **PGE Metadata (information about the PGE)**
  - **ESDT Metadata (information about each input/output of the PGE)**
  - **ORBIT Metadata (information about the orbit of the spacecraft supplying the data for the PGE)**
  - **TILE Metadata (information about the geographic tiles that the PGE will produce)**
  - **PATHMAP_ODL (information about mapping between Absolute Path Number and a Sequential numbering ranging from 1-233)**

Updating the PDPS Database with ESDT Metadata

- **PDPS needs basic information on every type of file associated with PGEs.**
  - **Metadata for the PDPS Database is first prepared in Object Definition Language (ODL), one ODL file for each ESDT.**
  - **Determine Science Data Server ShortName for ESDT corresponding to file.**
  - **Search ESDT directory for ESDT ODL file. If file exists, there is no need to make another one for this ESDT. The file naming convention is ESDT_*ShortName*.odl, where *ShortName* is the same name used for the Science Data Server.**
  - **If not, copy the ESDT ODL template from configured area to user space.**
  - **Add required metadata to ODL file via text editor.**
  - **ShortName in ODL file must match ShortName of file itself, ShortName in ESDT descriptor, and ShortName in PGE metadata ODL file.**
  - **Copy the ESDT ODL file to the configured area.**

Updating the PDPS Database with PGE Metadata

- **PDPS needs basic information or metadata on the PGE.**
  - **Order for this Update PGE Metadata Activity: It is recommended that all ESDT metadata ODL files associated with the PGE be prepared and put into the configured area before this procedure is executed.**
  - **Invoke PCF ODL Template Tool.**
  - **This tool prompts the user for the following information:**
    - **Configuration file - use default ConfigFile path and filename for DAAC**
    - **Process Control File - Path and filename of PCF, default path is current**
    - **PGE Name - Name of PGE associated with PCF**
    - **PGE Version - PGEversion, default needs to be determined if not known**

Updating the PDPS Database with PGE Metadata (Cont.)

- **Many PCF ODL files can be made on same invocation until user quits.**
- **Program outputs a file with name PGE_*PGEname#PGEversion*.tpl.**
- **Go to full path from which SSIT Manager is run and change this output ".tpl" filename extension to ".odl" or copy template into file with same name and ".odl".**
- **Edit PGE_*PGEname#PGEversion*.odl file to add metadata.**

Updating the PDPS Database with PGE Metadata (Cont.)

- **Process Steps:**
  - **From SSIT Manager select Tools → PDPS Database → SSIT Science Metadata Update**
  - **The program prompts for the following information:**
    - **ConfigFile - use default for path and filename at each DAAC.**
    - **MODE - use default of ops.**
    - **PGE name - name of PGE that will be registered (user choice).**
    - **PGE version - PGE version to be registered.**
  - **Quit out of program.**

Updating the PDPS Database with Operational Metadata

- **Assumption:**
  - **All ESDT metadata ODL files associated with the PGE must already be prepared and put into the configured area and the PGE must be registered using the Science Metadata Update Tool.**
- **Process Steps:**
  - **Invoke the PDPS/SSIT Database Update GUI Tool.**
  - **Select the PGE name and version.**
  - **Select new PGE and Done. PDPS needs basic operational metadata on the PGE to plan resources.**
  - **Select Profile and enter Performance Statistics values :**
    - **Wall clock time** **CPU time**
    - **Max memory used** **Block input ops**
    - **Block output ops** **Swaps**
    - **Page faults**

Updating the PDPS Database with Operational Metadata (Cont.)

- **Process Steps (Cont.):**
  - **Select Resource Requirements and enter values: Max disk space used during PGE run.**
  - **Select Proc. String. Only one should be listed. Number of Processors should be 1.**
  - **Select Apply to update the PDPS database.**
  - **To start over, use RESET button.**
- **The performance statistics collected by running the PGE under EcDpPrRusage will be used to initialize the PDPS database so that the Planning for DPRs can be performed.**
- **The actual values entered can be approximate.**

Placing Dynamic Data Granules on the Science Data Server

- **Dynamic test data granules are delivered with the PGE for input at runtime.**
- **Insert of dynamic test data is done by a Science Data Server Program.**
- **DAPs contain MCF templates or samples for output products, not input.**
- **PGEs, through the SDP Toolkit, generate target MCFs for each data granule produced using the source MCF ODL files delivered with the PGE.**

Placing Dynamic Data Granules on the Science Data Server (Cont.)

- **The target MCFs produced by the PGE are used by the Data Server to insert data products, thus they are called database load ODL files.**
- **SSI&T will have to create target MCFs (database load ODL files) for input test data granules since they are not produced by a PGE.**

Creating a Target MCF for a Dynamic Data Granule

- **Assumption:**
  - **All ESDT metadata ODL files associated with the PGE must already be prepared and put into the configured area and the PGE must be registered using the SSIT Science Metadata Update Tool.**
- **Process Steps to place dynamic data granules on the Science Data Server:**
  - **If a source MCF template is not available for the dynamic data granule, then make one by editing a template provided for SSI&T.**
  - **[continued]**

Creating a Target MCF for a Dynamic Data Granule

- **Process Steps to place dynamic data granules on the Science Data Server (Cont.):**
  - **Creating a target MCF for dynamic data granule from the source MCF.**
    - **Go to the directory where the source MCF resides.**
    - **Invoke SrcToTargetMCF program from command line.**
    - **Enter source MCF filename (.mcf) and target MCF filename (.met).**
    - **Edit TargetMCFfilename.met file.**
  - **For all Data_Location="PGE" attribute in Source MCF, enter data values.**
  - **Date values are provided by Instrument Teams on delivery of the DAP.**
    - **Data granule start date/time and end date/time must be provided.**
  - **Save the file.**

Inserting Dynamic Data Granules to the Science Data Server

- **Process Steps for inserting Dynamic Data Granules into the Science Data Server:**
  - **Go to the directory where the dynamic data granule resides.**
  - **From the SSIT Manager GUI → Tools → Data Server → Insert Test Dynamic**
  - **The Insert Test Dynamic program will be running.**
  - **Enter the following information:**
    - **ConfigFileName - use default.**
    - **ESDT ShortName - ESDT ShortName corresponding to data granule.**
    - **Filename to Insert - Full path & filename of data granule to be inserted.**
    - **Associated ASCII metadata (target MCF) filename to Insert - same as above filename but with ".met" extension.**

Placing Static Data Granules on the Science Data Server

- **Static data granules are those whose temporal locality is static over long periods of time.**
  - **Examples are calibration files which change only with a new version of the PGE.**
- **Static test data granules are delivered with some PGEs.**
- **Source MCFs delivered with PGEs are treated as static data granules.**
  - **Source MCFs delivered with PGEs can be used from the command line to test a PGE, however an MCF will have to be generated from the SDSRV for use in full PGE integration.**
- **Insert of static test data is done by a Data Server Program in SSIT Manager.**

Placing Static Data Granules on the Science Data Server (Cont.)

- **DAPs contain MCF templates only for output products, not for input files.**
- **PGEs, through the SDP Toolkit, generate target MCFs for each data granule produced using the MCF files generated by the SDSRV.**

Placing Static Data Granules on the Science Data Server (Cont.)

- **Target MCFs (database load ODL files) are needed to insert all data files, including static data granules, to the Science Data Server.**
- **SSI&T will have to create target MCFs (database load ODL files) for static data granules delivered for SSI&T since they are not produced by a PGE.**
- **A template for static Metadata ODL files is provided for SSI&T.**
- **The static version has parameters unlike those for dynamic data granules.**

Creating a Metadata ODL File for a Static Data Granule

- **Assumption:**
  - **All ESDTs associated with the PGE, including either a single ESDT for static type files or an ESDT for each static file, must already be registered. PGE must be registered using Science Metadata Update Tool.**
- **A template metadata ODL file which can be edited is provided for SSI&T by the instrument team.**

Creating a Metadata ODL File for a Static Data Granule

- **Process Steps to place static data granules on the Data Server:**
  - **Create a metadata ODL file for a static data granule.**
    - **Go to the directory where the metadata ODL template resides.**
    - **Copy StaticODLmet.tpl to the user work directory as filename.met.**
    - **Edit the filename.met and enter the following information and save:**
      - **ShortName - ESDT ShortName**
      - **VersionID - ESDT VersionID**
      - **ParameterName - name of static file in InformationContentContainer**
      - **ParameterValue - Cn: C = coefficient file, M = MCF; n = 1,2,...**

Inserting Static Data Granules to the Data Server

- **Process Steps for inserting Static Data Granules to the Data Server:**
  - **Go to the directory where the dynamic data granule resides.**
  - **From the SSIT Manager GUI → Tools → Data Server → Insert Static**
    - **The Insert Static program will be running.**
  - **Enter the following information:**
    - **Config Filename - use default.**
    - **Mode - use default mode "ops"**
    - **ESDT ShortName - ESDT ShortName for data (bucket ESDT).**
    - **Science Group - Cn: C = coefficient file, M = MCF; n = 1,2,…**
    - **PGE Name - Name of PGE registered.**

Inserting Static Data Granules to the Data Server (Cont.)

- **Process Steps for inserting Static Data Granules to the Data Server (Cont.):**
  - **Enter the following information:**
    - **PGEVersion - PGE version or take default of 1.**
    - **Filename to Insert - Filename of static data granule to be inserted.**
    - **Associated ASCII metadata (target MCF) filename to Insert - same as the above filename but with ".met" extension.**
  - **The static granule will be inserted. The program can run again until the user quits.**

Inserting Science Software Executable on Science Data Server

- **All science software executables must be inserted on the Data Server to be run by the PDPS**
  - **The executable package is called a SSEP or EXE Tar**
  - **The steps of this procedure:**
    - **Make a new directory to hold the contents of the SSEP.**
    - **Copy all files to go into the SSEP into this directory: PGE executables shell scripts, SDP Toolkit message files.**
    - **Use UNIX tar to make the package**
    - **Copy over the Target MCF template to filename.met, edit and save:**
    - **The PGE name is PGEEXE. Enter PGE version and parameter values according to program prompt.**

Inserting Science Software Executable on Science Data Server

- **All science software executables must be inserted on the Data Server to be run by the PDPS (Cont.)**
  - **The steps of this procedure (Cont.):**
    - **Select from the SSIT Manager GUI: Tools → Data Server → Insert EXE TAR.**
    - **Enter PGE Name, SSWVersion, SSEPFileName, ExecFileName**

PGE Planning and Processing

- **After the PGE has been linked to the DAAC Version of the SDP Toolkit, all associated ESDT and PGE information has been entered into the PDPS Database, all operational metadata has been entered, and the PGE has been registered, then the PGE is ready to be run in PDPS under AutoSys.**

PGE Planning and Processing

- **The major steps in the PGE Planning and Processing are the following:**
  - **Register a subscription for test input and output files (once per ESDT).**
  - **Using the Production Request Editor, enter processing information and submit a Production Request.**
  - **Using the Planning Workbench, create a new production plan, schedule the processing, activate the plan, and review the planning timeline.**
  - **Monitor production under AutoSys.**
  - **Using the QA monitor, acquire and view the test output file from the Data Server and examine the Production History File.**

Registering Subscriptions for Test Input and Output Files

- **The Data Subscription Management application is used to manage the receipt of a subscription notification from the Data Server.**
- **Process Steps:**
  - **Invoke the Subscription Editor from the SSIT Manager.**
  - **From the SSIT Manager GUI: Tools → Data Server → Register Subscription**
  - **Register a subscription for each of the input files and output product files associated with the PGE.**

PGE Planning, Processing, and Product Retrieval

- **Production planning and processing activities are essential elements of SSI&T**
- **Procedures for managing production requests, production plans, and production processing are described in detail in 625-EMD-006, Training Material for the EMD Project Volume 6: Production Planning and Processing, and have not been duplicated in this lesson**

Post Processing Activities

- **Invoke the QA Monitor to view the products and select options.**
- **Select Query data and select or query on ESDT and Duration (time range).**
  - **Select Data Granule(s) to check and select Retrieve Data Granule.**
  - **Production History Log files from PGEs run within PDPS.**
  - **Find Data Granule and select Retrieve Production History.**
  - **Examine Production History Log for metadata and other information.**

Post Processing Activities

- **EOSView**
  - **Select Visualize Data.**
  - **Select Attributes to examine metadata in HDF headers.**
  - **Select Table to view data vectors and matrices.**
  - **If an image file is available, select Visualize.**
  - **From the SSIT Manager GUI → Tools → Product Examination → File Comparison → EOSView [or IDL]**

Postprocessing and General Investigation

- **Detection of Science Software Problems and Production System Problems**
  - **Examine PGE Log Files**
  - **Investigate errors in Production History File**
  - **Use File comparison tools, EOSView and IDL to examine data product problems**
  - **Examine the PDPS Database with the Web Browser**

File Comparison and Data Visualization

- **An important part of SSI&T is verifying that the output files produced at the DAAC are identical (within particular tolerances) to the test output files delivered with the DAPs**
- **A successful comparison is a strong indication that the porting of the science software from the development facility at the SCF to the operational facility at the DAAC has not introduced any errors**

File Comparison and Data Visualization (Cont.)

- **A number of file comparison tools are available during SSI&T via the SSIT Manager GUI or they can be invoked from the UNIX command line**
  - **Two tools are available for comparing HDF or HDF-EOS files**
    - **HDF File Comparison GUI**
    - **hdiff HDF File Comparison Tool**
  - **One tool is available for comparing ASCII files**
    - **ASCII File Comparison Tool**
  - **Another tool is available for assisting in comparing binary files**
    - **Binary File Difference Assistant**

File Comparison and Data Visualization (Cont.)

- **Data Visualization**
  - **In order to view the success of science software in producing scientifically valid data sets, the data needs to be displayed in forms that convey the most information**
    - **Data visualization enables this to be done**
  - **There are two visualization tools provided to the DAAC**
    - **EOSView**
    - **Interactive Data Language (IDL)**
  - **These tools are both accessible via the SSIT Manager**

File Comparison and Data Visualization (Cont.)

- **Data Visualization (Cont.)**
  - **EOSView is user friendly GUI for creating two-dimensional displays from HDF-EOS objects (Grid, Swath) as well as the standard HDF objects (SDS, Vdata, Image, Text)**
    - **It has additional features such as thumbnail-panning, colorization, zooming, plotting, and animation**
  - **IDL is a COTS display and analysis tool used to create two-dimensional, three dimensional (volumetric), and surface/terrain displays from binary, ASCII, and many other formats in addition to HDF**

Miscellaneous

- **Setting up an environment for direct PDPS database access**
  - **The PDPS database contains many tables containing information about science software running in the production system**
  - **The population of some of this information is part of standard SSI&T procedures and no special environment set-up is required for these procedures**
  - **It may be necessary, however, to gain direct access to the PDPS database from time to time and this procedure describes how to set up the required environment**
  - **Setting up an environment for direct PDPS database access involves editing and "sourcing" a .cshrc file**