



Build PGE Binaries for Testing

Tom Atwater

520-TD-001-002 SSI&T 6-1

Discussion Topics

Now that we have tested the science software for ANSI compliance and made sure that there are no prohibited function calls, the next step is to make sure that it builds properly.

Since the software was developed at the SCF using the SCF version of the SDP toolkit, we want to make sure that it gives the same output.

Build PGE Binaries for Testing



- Compiling a PGE and linking with SCF version of SDP Toolkit
- Comparing output with the one provided by SCF
- Compiling status messages file
- Extracting prologs from science software source files

520-TD-001-002

SSI&T 6-2

Discussion Topics

First we want to compile the code at the DAAC and link to the SCF version of the SDP toolkit. If everything is OK, we run the compiled program. We should have the same results as generated at the SCF, within a given tolerance.

Compiling a PGE and Linking with SCF Version of SDP Toolkit



Purpose:

- To compile a delivered software and link it with SCF version of SDP Toolkit

Tool:

- Install scripts, build scripts, make files

Assumptions:

- The science software delivery has been unpacked (e.g. un-tarred) and placed into the software build area. All necessary files are available, accessible, and have the proper permissions set. The build process will be done on the SGI Power Challenge in a Unix shell

520-TD-001-002 SSI&T 6-3

Discussion Topics

Purpose: To compile delivered science software and link it with the SCF version of the SDP Toolkit.

Assumptions: The science software delivery has been unpacked (e.g. un-tarred) and placed into the software build area. All files necessary to build the science software are available, accessible, and have the proper permissions set. This includes any install scripts, build scripts, make files, and instructions for using them. The build process will be done on the SGI Power Challenge in a Unix shell (that is, outside of CASEvision or other COTS environments).

References:

Delivered System Description Document and Operations Manual from Instrument Team

SDP Toolkit User's Guide

NOTE: The compiling and linking of science software (building) will vary according to the particular delivery. The instructions supplied with the delivery should be the primary source of information. What follows below are some "typical" steps that may or may not be applicable to a particular situation.

Compiling a PGE and Linking with SCF Version of SDP Toolkit (cont'd)



- Read all instructional information supplied with the delivery
- Start a new shell on the SGI science processor
- Set PGSHOME appropriately
- Set up the environment within which to do software builds
- Examine and alter (if necessary) any supplied make files
- Compile any Status Message Files
- Verify that directory structure is as intended
- perform the build according to instructions

520-TD-001-002 SSI&T 6-4

Discussion Topics

Step 1. Read all instructional information supplied with the delivery.

- Such material should be the primary source of information on how to build science software.
- Use any appropriate viewer or editor desired.
 - ASCII (text) files may be viewed with more or through vi.
 - PostScript documents may be viewed through ghostview and
 - PDF documents may be viewed through Acrobat Reader (both accessible via the SSIT Manager).
 - Documents in Microsoft Word and related formats may be viewed through Microsoft Word (available via the SSIT Manager).

Step 2. Start a new shell on the SGI science processor within which to build the software.

Either:

- From the SSIT Manager GUI, choose the Tools menu.
 - Then choose Xterm.
 - Then telnet into the SGI.
- or
- In any currently available Xterm window, spawn a new Xterm session with: xterm &
 - Then telnet into the SGI.

NOTE: Starting a new shell avoids potential problems associated with redefinitions of environment variables.

Compiling a PGE and Linking with SCF Version of SDP Toolkit (cont'd)



- Read all instructional information supplied with the delivery
- Start a new shell on the SGI science processor
- Set PGSHOME appropriately
- Set up the environment within which to do software builds
- Examine and alter (if necessary) any supplied make files
- Compile any Status Message Files
- Verify that directory structure is as intended
- perform the build according to instructions

520-TD-001-002 SSI&T 6-5

Discussion Topics

Step 3. Set PGSHOME appropriately.

- Enter setenv PGSHOME pathname

where pathname is the home directory of the appropriate SCF version of the SDP Toolkit. There may be multiple versions of the SCF version of the SDP Toolkit depending on the mode the software is built in (old 32-bit, new 32-bit, or 64-bit) and on the whether the software includes FORTRAN 77 or Fortran 90.

NOTE: Make sure that PGSHOME points to the correct SCF version.

Step 4. Set up the environment within which to do software builds.

Source the appropriate pgs-dev-env.csh or pgs-dev-env.ksh file:

- source \$PGSBIN/pgs-dev-env.csh
- or
- source \$PGSBIN/pgs-dev-env.ksh

NOTE: The environment variable PGSHOME must be set first.

The choice of .csh or .ksh depends on your current shell. Use the .csh file if you are running C shell, .ksh if you are running Korn shell.

If you don't know, within the xterm window enter, echo \$SHELL

Step 5. Examine and alter (if necessary) any supplied make files.

- Invoke favorite text editor (vi, emacs, xedit, etc.)
- Check that compiler, compiler flag settings and other environment variable settings are appropriate. Try to use the same settings as the SDP Toolkit.
- Assuming step 4 was done, use the command env to list current settings of the shell's environment variables.

Compiling a PGE and Linking with SCF Version of SDP Toolkit (cont'd)



- Read all instructional information supplied with the delivery
- Start a new shell on the SGI science processor
- Set PGSHOME appropriately
- Set up the environment within which to do software builds
- Examine and alter (if necessary) any supplied make files
- Compile any Status Message Files
- Verify that directory structure is as intended
- perform the build according to instructions
- Move executables out of ClearCase
- Run the PGE

520-TD-001-002 SSI&T 6-6

Discussion Topics

Step 6. Edit PCF to replace SCF directory paths with DAAC ones

NOTE: The delivery may include several versions of make files appropriate to different platforms. Refer to supplied documentation.

The delivery may include install scripts which set environment variables.

Try to make use of environment variables used by the shell (e.g. F77, C, CFLAGS) rather than setting them within the make/build files.

Step 7. Compile any Status Message Files

- Run smfcompile.
- Move the text message files into the \$PGSMSG directory.
- Move the include files into the directory expected by the build/make files.

NOTE: With some deliveries, the build scripts may perform this step automatically.

See section on Compiling System Message Files (SMFs) for details on how to do this.

Step 8. Verify that directory structure is as intended.

NOTE: Deliveries may come with install scripts which place files into various directories according to some structure.

Step 9. Once installation scripts, make files and build scripts are setup properly, perform the build according to instructions.

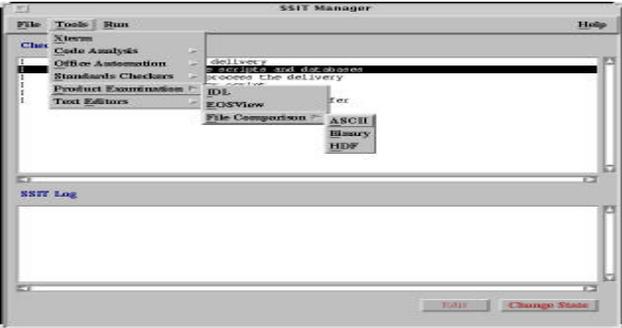
NOTE: Science software deliveries may come with a single, top-level script to do the entire build or the build process could involve a series of steps, each of which should be described fully in the delivered documentation.

Step 10: Run the PGE

The output that's generated will then be compared to the output sent by the SCF.

Compare PGE Outputs

- Comparing Two ASCII Files
- Comparing Two Binary Files
- Comparing two HDF Files



The screenshot shows the SSI Manager application window. The menu bar includes File, Tools, Run, and Help. The 'Tools' menu is open, showing options like Code Analysis, Office Automation, Standards Checkers, Product Examination, and Text Editors. The 'File Comparison' option is selected, and a sub-menu is displayed with options for ASCII, Binary, and HDF. The main workspace is empty, and there is an 'SSIT Log' section at the bottom. The window title is 'SSIT Manager'. The bottom left corner of the slide contains the text '520-TD-001-002' and the bottom right corner contains 'SSI&T 6-7'.

Discussion Topics

One of the critical things that we need to do in testing the science software is to compare the PGE file output generated at the SCF to the output generated at the DAAC.

There are three potential file comparisons to perform

- ASCII files
- binary files
- Hierarchical Data Format (HDF) files

Within certain predetermined tolerance levels, the comparison should yield similar results for the DAAC-run software and the SCF-run software, using the SCF version of the SDP toolkit.

If the results are not similar, then something is clearly wrong. This is when we contact the SCF.

In the next few slides we'll talk about how to perform file comparison using the three formats.

Comparing Two ASCII Files



Purpose:

- To Determine if the SCF and the DAAC ASCII files are identical

Tool:

- SSIT Manager, ASCII files comparison utility

Assumptions:

- ASCII File from SCF is available, Output file generated at the DAAC

520-TD-001-002 SSI&T 6-8

Discussion Topics

Purpose:

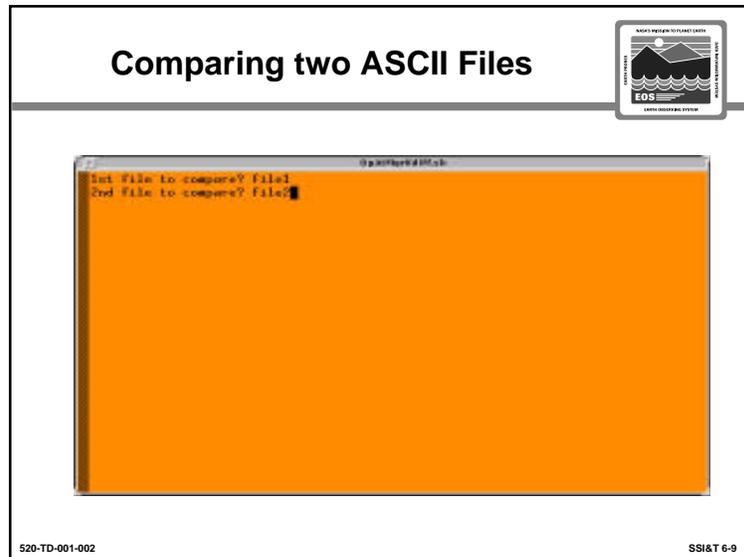
- To Determine if the SCF and the DAAC ASCII files are identical

Tool:

- SSIT Manager, ASCII files comparison utility

Assumptions:

- ASCII File from SCF is available, Output file generated at the DAAC using the SCF version of the SDP toolkit



Discussion Topics

Step 1. From the SSIT manager:

- select "Tools"
- select "File Comparison"
- select ASCII

Step 2. At the prompt:

- enter the name of the first file and press <CR>
- then enter the name of the second file

NOTE: Use full path name for each file.

Step 3. compare the two outputs for differences

NOTE: If no window appears, this means the files are identical. SUCCESS!

Comparing Two Binary Files



Purpose:

- Determine if two binary files are the same or different

Tool:

- SSIT Manager, Binary File Differences Assistant

Assumptions:

- Binary File from SCF is available, Output file generated at the DAAC

520-TD-001-002 SSI&T 6-10

Discussion Topics

Purpose:

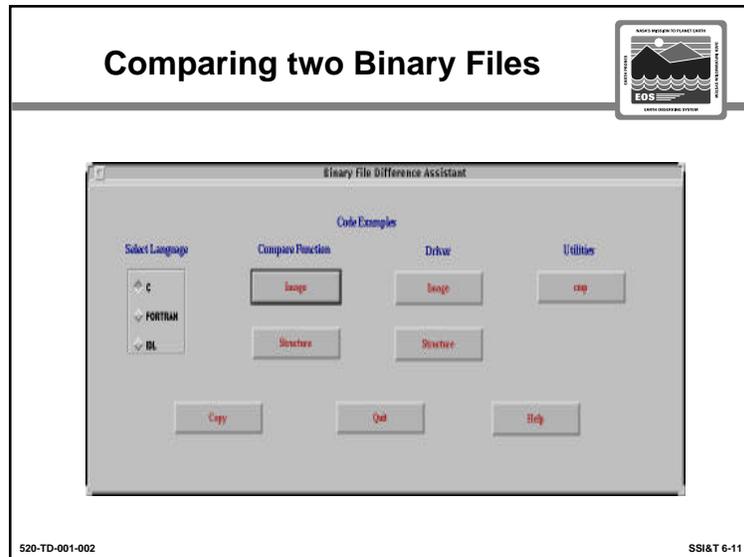
- To Determine if the SCF and the DAAC binary files are identical

Tool:

- SSIT Manager, Binary File Differences Assistant

Assumptions:

- Binary file from SCF is available, Output file generated at the DAAC using the SCF version of the SDP toolkit



Discussion Topics

Step 1. From the SSIT manager:

- select “Tools”
- select “File Comparison”
- select Binary

Step 2. Select the language (c, FORTRAN, IDL)

- decide to view or copy code to disk

If you select to view the code

- select one of the following:
 - Compare Function
 - Driver
 - Utilities

If you select to copy the code to disk

- select sample code to write to disk

A window asks you for a “File Identifier”

- identify your code
- customize your code by editing it to conform to your binary structure

Comparing Two HDF Files



Purpose:

- Determine if two HDF files are the same or different (within a predetermined tolerance level)

Tool:

- SSIT Manager, HDF file comparison tool

Assumptions:

- HDF File from SCF is available, Output file generated at the DAAC

520-TD-001-002 SSI&T 6-12

Discussion Topics

Purpose:

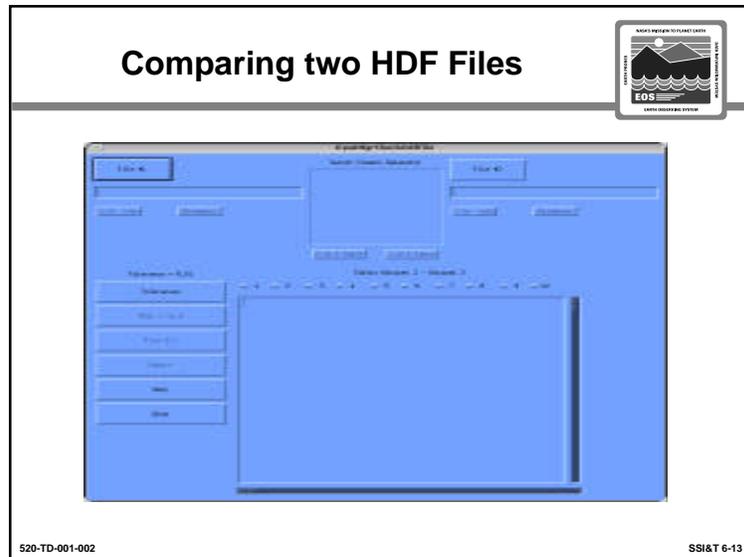
- To Determine if the SCF and the DAAC HDF files are identical (within predetermined tolerance level)

Tool:

- SSIT Manager, HDF File comparison tool

Assumptions:

- HDF file from SCF is available, Output file generated at the DAAC using the SCF version of the SDP toolkit



Discussion Topics

Step 1. From the SSIT manager:

- select "Tools"
- select "File Comparison"
- select HDF

Step 2. Load first file

- select file 1

NOTE: start with the path name, and enter it segment by segment

e.g., /home <CR>
 /first subdirectory <CR>
 /second subdirectory <CR>
 etc.

- set the filter to "*.hdf"
- select file 1 to be compared

Step 3. Load second file (same as step 2)

Step 4. select the desired number of data sets (in the center window)

Step 5. select Report to view the results in a report

Compiling Status Messages File



Purpose:

- To compile Status Message File(s) (SMFs) delivered with science software into message files and include (header) files. These files will then be used by the science software during runtime

Tool:

- smfcompile, PGMSG and PGSINC environment variables

Assumptions:

- The delivered SMFs are available, accessible, and have read permissions. The program smfcompile (included with the Toolkit) is accessible. The environment variables PGMSG and PGSINC are properly set.

520-TD-001-002 SSI&T 6-14

Discussion Topics

Purpose: To compile Status Message File(s) (SMFs) delivered with science software into message files and include (header) files. These files will then be used by the science software during runtime.

Assumptions: The delivered SMFs are available, accessible, and have read permissions. The program smfcompile (included with the Toolkit) is accessible. The environment variables PGMSG and PGSINC are properly set.

References:

SDP Toolkit User's Guide for the ECS Project, 333-CD-003-002

The Toolkit Primer, available through the World Wide Web,

URL: <http://newsroom.hitc.com/sdptoolkit/primer/tkprimer.html>

Compiling Status Messages File (cont'd)



- Start a new shell on the SGI science processor
- Switch to directory containing status message files
- Run smfcompile on the status message files
- Place the created include files and the created runtime ASCII files into proper directories
- Repeat as necessary

520-TD-001-002 SSI&T 6-15

Discussion Topics

Step 1. Start a new shell on the SGI science processor within which to run smfcompile.
Either:

- From the SSIT Manager GUI, choose the Tools menu. Then choose Xterm. Then telnet into the SGI within the xterm.

or

- In any currently available Xterm window, spawn a new Xterm session with: `xterm &`
- Then telnet into the SGI within the xterm.

Science software will be run on the SGI. Therefore, compiling status message files should be done on the same machine.

Step 2. Go into the directory containing the status message files.

- Use `cd` to change directories.

NOTE: The user can be in any other directory while running smfcompile, however, full pathnames of the input files will have to be specified.

Step 3. Run smfcompile on the status message files.

Enter,

- `smfcompile -f textfile`

or

- `smfcompile -r -i -f textfile`

where textfile is the filename of the status message text file delivered with the science software.

Compiling Status Messages File (cont'd)



- Start a new shell on the SGI science processor
- Switch to directory containing status message files
- Run smfcompile on the status message files
- Place the created include files and the created runtime ASCII files into proper directories
- Repeat as necessary

520-TD-001-002 SSI&T 6-16

Discussion Topics

NOTE: Typically, the delivered status message textfiles should have filename extensions of .t and have the seed value of used by the textfile as part of the filename, (e.g. PGS_MOD_39123.t, where 39123 is this message file's seed number).

Use the -r option to redirect the ASCII runtime message file to the directory set in the PGSMMSG environment variable. Use the -i option to redirect the include file to the directory set in the PGSINC environment variable.

Only one textfile at a time can be processed; wildcards cannot be used.

Step 4. Place the created include files and the created runtime ASCII files into the proper directories.

- Enter `mv include_filename $PGSINC`
- `mv runtime_filename $PGSMMSG`

where include_filename is the name of the newly created include file and runtime_filename is the name of the newly created runtime ASCII message file.

NOTE: This step is unnecessary if both the -r and -i options were used in the previous step.

Step 5. Repeat as necessary.

- Repeat steps 1-4 as necessary for each of the status message files associated with the science software.

NOTE: Only one textfile at a time can be processed; wildcards cannot be used.

Extracting Prologs from Science Software Source Files



Purpose:

- To extract prologs from science software source file modules

Tool:

- SSIT Manager

Assumptions:

- The science software source files are available, accessible, and have read permissions

520-TD-001-002
SSI&T 6-17

Discussion Topics

Purpose: To extract prologs from science software source file modules. By default, these prologs are assumed to be surrounded by standard delimiters that comply with Data Production Software and Science Computing Facility (SCF) Standards and Guidelines (CH-01, 2/15/95, NASA/ESDIS, GSFC, Code 505, Greenbelt, MD).

Tools/Materials/Assumptions: The science software source files are available, accessible, and have read permissions. The starting prolog delimiters are one of:

Language	Type	Delimiter
FORTRAN 77	source	!F77
Fortran 90	source	!F90
C	source	!C
Ada	source	!ADA
FORTRAN 77	include	!F77-INC
Fortran 90	include	!F90-INC
C	include	!C-INC

The end delimiter is always !END.

The source files to be examined have filename extensions which are one of:

FORTRAN 77:	f, f77, ftn, for, F, F77, FTN, FOR
Fortran 90:	f90, F90
FORTRAN 77/Fortran 90 include:	inc, INC
C:	c
C/C++ header:	h
Ada:	a, ada

The output filename will be prologs.txt (this can be overridden by changing its reference in the SSI&T Manager's Process Control File or with command-line options; see references below).

The location of the output file will be in the directory from which the SSI&T Manager is being run.

References:

Interim Release One (Ir1) Maintenance and Operations Procedures, 609-CD-001-001

Help screen available through SSI&T Manager (from SSI&T Manager, choose Help)

Data Production Software and Science Computing Facility (SCF) Standards and Guidelines, CH-01, 2/15/95, NASA/ESDIS, GSFC, Code 505

Extracting Prologs from Science Software Source Files (cont'd)



- Invoke the prolog extractor
- Specify the filenames of the source files to be examined
- View the contents of the desired file

520-TD-001-002 SSI&T 6-18

Discussion Topics

NOTE: The eventual goal of prolog extraction is TBD.

Step 1. Invoke the Prolog Extractor.

- From Tools menu of the SSI&T Manager choose Standards Checkers.
- Then choose Prolog Extractor.

An xterm (on the Sun) will be displayed within which the Prolog Extractor is being run.

Step 2. Specify the filenames of source files to be examined.

- To the prompt: File(s)? (-h help)
- list files and/or directories containing the files.

The "current directory" is the directory from which the SSI&T Manager was run.

- Separate filenames with spaces.

NOTE: If directories are specified (and they exist within the current directory), their contents will be searched recursively for files with valid filename extensions.

All output is appended to the same output file.

You can list './' as a short-hand notation for all valid files within the current directory and all directories below. (Note that the slash is necessary; using just '.' will not work.)

The time needed for the Prolog Extractor could be very large for large numbers of files and directories.

Step 3. If desired, view the contents of the output file.

- Use any text viewer or editor desired.

Step 4. Quit the Prolog Extractor

- To the prompt, enter q The xterm session running the Prolog Extractor will disappear.