CTN/MESA Test Tools

A Guide to Programs for Testing DICOM/HL7 Functions

Stephen Moore
Mallinckrodt Institute of Radiology
510 S. Kingshighway Blvd.
St. Louis, MO 63110
Voice: 314.362.6965
Fax: 314.362.6971

CTN Version 3.0.3
MESA Version 5.5.3
April 17, 2002

Contents

1 Introduction

This manual describes several utility programs that have proved to be useful during the development of the CTN software. The style adopted for most tests is to write simple, short programs that exercise or demonstrate specific parts of the system. Thus, there is not one general application whose purpose is to exercise the entire system.

Most of the utilities are written to accept a number of command line arguments (and switches). That is, these applications do not prompt your for parameters, nor do they operate in an interactive mode. All of the applications will specify what arguments are needed if they are invoked with no arguments (or an insufficient number of arguments). This short listing of arguments is intended for someone who is already familiar with the application and is not expected to replace the written documentation in this manual.

These documents are not intended to explain the implementation details of the utilities. Many of the details can be inferred from the function of the utility, and the documentation present in the source code for the individual utilities will aid in understanding the design and implementation details.

Starting with the December 2001 version of this document, we now combine the MIR CTN test tools and MESA test tools in a single document. Although these tools are maintained as separate projects, there are available as a whole if you install the MESA software. CTN users may find the added tools helpful and choose to migrate to the MESA package.

2 DICOM Object Test Programs

This section describes utilities that are useful for examining DICOM Information Objects (images) that are stored in files. This software assumes that Information Objects are stored in files as a stream of bytes that correspond to the DICOM Little-Endian Transfer Syntax. (The file format described in Part 10 of the Standard is not yet implemented.) The programs described in this section can be used to examine the contents of these files. dcm_dump_file reads a file and prints an ASCII description of each element that is present. dcm_dump_element extracts a single element and writes the data in binary to another file (very useful for extracting pixel data). dcm_verify examines a file and determines which Information Entities, Modules and Attributes are present per the tables defined in Part 3 of the Standard.

3 General Queue Test Programs

The queuing facility was extended for DICOM `93 to accept a user-defined queue element which is defined at run time. As such, only a few utilities exist since this data structure must be correctly defined for whatever queue is being used. However, two utilities exist for manipulating queues. gqinitq and gqkillq will (respectively) create a new queue and destroy an existing queue. In the ../apps/gq directory the source to gqtest.c shows gives some examples of working with other queue functions.

4 A Display Utility for DICOM Image Files

dcm_x_disp is an X-based display utility for DICOM image files. It is fairly simple but useful since it uses exactly the same algorithm for displaying image data as *ctndisp*. Therefore, display problems witnessed with ctndisp can be verified with *dcm_x_disp* to help find the source of any problems. *dcm_x_disp* also dumps out useful information about the image after displaying it.

The program *dcm_w_disp* is a display program in the windows environment. It provides slightly less functionality than *dcm_x_disp*.

5 Print Management Functions

print_client is a program that requests applications with print servers and sends preformatted images to be printed. This client program does not have a GUI and is intended as a mechanism to test the protocol for printing images. print_client is not designed to be a general purpose program for print images. Please refer to the documentation for print_manager found in *User's Guide for CTN Demonstration Applications* for a general purpose print manager with a GUI.

6 Example Protocol Data Units

There are several applications that are written to generate association request and response messages and to dump the binary data in hexadecimal (ASCII) to be used as a learning tool. These applications are collectively described on the page labeled pdus.

7 Platforms

The CTN and MESA software is supported on Unix and Windows NT platforms. Not all applications are available on both platforms. Each program description to follow will indicate which platform supports the application.

8 CTN Utilities

8.1 dcm_add_fragments

Name

dcm_add_fragments – add one or more encapsulated pixel fragments to a DICOM composite object

Platforms

Unix, Windows

Synopsis

```
dcm_add_fragments [-t] [-x xfer] filein fileout f1 [f2…]
```

Description

dcm_add_fragments opens an existing DICOM file and adds one or more fragments to the file. The fragments replace any existing pixel data, so the application does not append fragments. The application assumes the user has performed any required compression to the fragments.

The application also assumes the user knows the UID of the transfer syntax that describes the compression scheme that has produced the fragments. If the user specifies an input file that is in Part 10 format and has the proper transfer syntax UID, then the user can merely replace the fragments. If the user specifies an input file that is not in Part 10 format or that uses a different transfer syntax, then the –x switch is required to specify the proper transfer syntax for the output file.

Options

-t	Input file is in DICOM Part 10 format (default is CTN format)
-x	Specify the UID of the transfer syntax of the output file
filein	Input file name
fileout	Output file name
f1	First encapsulated fragment
f2	Zero or more fragments to add (in order) after f1

Notes

8.2 dcm_create_object

Name

dcm_create_object – create a DICOM object file from text input

Platforms

Unix, Windows

Synopsis

```
dcm_create_object [-i inputfile] [-p pixels] [-v] fileout
```

Description

dcm_create_object reads a text stream from the standard input or the file specified with the –i switch and creates a DICOM object. This object contains only those attributes specified by the user; there is no guarantee by this program that the output is a "legal" DICOM composite object, a DICOM normalized object, or any other type of DICOM object. The tool merely provides a translation mechanism.

The user may also add pixel data with the –p switch. The file format of pixels is raw data. We do not support compressed data with this application.

The input stream follows the CTN Advanced Object Specification described in Appendix A of this document.

Options

-i	Take input from this file rather than stdin
-p	Read raw pixel data from this file and add as 7FE0 0010
-v	Verbose mode

Notes

dcm_make_object is a similar program that uses a simpler input syntax.

8.3 dcm_ctnto10

Name

dcm_ctnto10 - Convert a file in "CTN" format to a format that conforms to Part 10 of the DICOM Standard.

Platforms

Unix, Windows

Synopsis

dcm_ctnto10 [-BL] [-v] filein fileout

Description

dcm_ctnto10 converts a file from the traditional CTN format to a file in DICOM Part 10 format. The program assumes the input transfer syntax is the DICOM default Little-Endian transfer syntax. The default output transfer syntax is also DICOM default Little-Endian transfer syntax. The user can specify DICOM explicit big-endian or DICOM little-endian transfer syntax.

The options are:

-B	Output file should be written in DICOM explicit big-endian transfer syntax
-L	Output file should be written in DICOM explicit little-endian transfer syntax
-v	Place DCM facility in verbose mode.

Notes

We have some problems with our encoding of pixel data with the big-endian transfer syntax, especially 8-bit data. Be wary of the output of this particular program in that regards. It is safest to stick with the little-endian transfer syntaxes for now.

8.4 dcm_diff

Name

dcm_diff - Compare the attributes in two files

Platforms

Unix, Windows

Synopsis

dcm_diff -b -g -l -o -t -v -z file1 file2

Description

dcm_diff reads the user designated files and compares the attribute values. Differences are printed to the standard output. Differences include the cases where an attribute is present in one file and missing in the other or the attribute values are different. This version of the program does not compare pixel data.

The options are:

-b	Input files are stored in big-endian byte order
-g	Remove group length elements
-l	Use (retired) length-to-end attribute for object length
-o	Place output in verbose mode
-t	Part 10 file
-v	Place DCM facility in verbose mode
-z	Perform format conversion (verification) on data in files

Notes

Comparing DICOM files is not a straightforward process. If you send a DICOM image to a storage SCP and then retrieve the image, you may find small differences in the header. For example, the storage SCP may coerce certain attributes (like the patient ID or the study UID). The storage SCP may also remove attributes that are not in its dictionary or may change the format of an attribute ("1.0" may become "1.000").

This program prints differences to the standard output and lets the user decide if the files are different. We did not feel it was appropriate to set a policy on what constituted a difference that would be considered significant in all applications.

8.5 dcm_dump_element

Name

dcm_dump_element - dump one element in binary from a DICOM information object

Platforms

Unix, Windows

Synopsis

dcm_dump_element [-b] [-t] [-v] group element filein fileout

Description

dcm_dump_element reads a DICOM information object from a file and extracts one data element. The binary data for that element is written to an output file. In the argument list, group and element should be specified in hex. filein and fileout specify the input and output files, respectively.

The options are:

-b	Read data from input file assuming big-endian format.
-t	Read data from input file assuming DICOM Part 10 format.
-v	Verbose mode. Turn on verbose mode for DCM facility.

Notes

This is the method that we use to extract pixel data from images.

See Also

Programmer's Guide to the DCM Facility

8.6 dcm_dump_file

Name

dcm_dump_file - dump the contents of a DICOM file to standard output in a text format.

Platforms

Unix, Windows

Synopsis

dcm_dump_file [-b] [-e] [-g] [-l] [-m mult] [-t] [-v] [-z] file [...]

Description

dcm_dump_file uses the DCM_DumpElements function to dump the contents of a DICOM file to standard output in a human-readable form. The information printed for each data element includes:

- Tag (Group and Element Number)
- Data value length
- Short english description
- Some or all of the data

The english description is found in the data dictionary in the DCM package. Data elements not in the dictionary will not have a description.

Some of the data from each element will be printed if the element is found in the data dictionary.

The following options are available:

-b	Read data from input files assuming big-endian format.
-e	Exit on file open error. Do not process other files. Normal mode is to continue processing other files.
-g	Remove group length elements as file is read.
-l	Use (retired) length-to-end attribute to find the end of the file.
-m mult	For binary attributes with vm > 1, print mult attribute values.
-t	Part 10 file
-v	Verbose mode. Turn on verbose mode for the DCM facility.
-z	Perform format conversion (verification) on data in files.

See Also

Programmer's Guide to the DCM Facility

8.7 dcm_make_object

dcm_make_object – create a DICOM object file from text input

Platforms

Unix, Windows

Synopsis

```
dcm_make_object [-p pixels] [-v] fileout
```

Description

dcm_make_object reads a text stream from the standard input and creates a DICOM object. This object contains only those attributes specified by the user; there is no guarantee by this program that the output is a "legal" DICOM composite object, a DICOM normalized object, or any other type of DICOM object. The tool merely provides a translation mechanism.

The user may also add pixel data with the –p switch. The file format of pixels is raw data. We do not support compressed data with this application.

The input stream follows the CTN Simple Object Specification described in Appendix A of this document.

Options

-p	Read raw pixel data from this file and add as 7FE0 0010
-v	Place DCM facility in verbose mode

Notes

dcm_create_object is a similar program that provides more features.

8.8 dcm_map_to_8

Name

dcm_map_to_8 - map original pixel data (10, 12 bit) to 8 bit

Platforms

Unix, Windows

Synopsis

dcm_map_to_8 [-b] [v] [-W width] [-C center] input output

Description

dcm_map_to_8 reads one DICOM image (monochrome) and maps the pixel data to 8 bits. It does this by applying window width, window center, rescale slope and rescale intercept values found in the information object. The user can override the window width and center values by supplying different values as command line arguments. The following options are available:

-b	Read input file assuming the data is stored in big-endian byte order
-v	Verbose mode. Place DCM facility in verbose mode.
-W width	Override window width with this value.
-C center	Override window center with this value.

Notes

We use this application to map data from modalities for use with our print client software.

8.9 dcm_modify_elements

Name

dcm_modify_elements - modify data elements in an existing DICOM information object

Platforms

Unix, Windows

Synopsis

```
dcm_modify_elements [-b] [-p pixels] [-t] [-v] filein fileout
```

Description

dcm_modify_elements reads an existing DICOM information object from a file and modifies individual data elements based on input read from stdin. This object contains only those attributes specified by the user; there is no guarantee by this program that the output is a “legal” DICOM composite object, a DICOM normalized object, or any other type of DICOM object. The tool merely provides a translation mechanism.

The user may also modify pixel data with the –p switch. The file format of pixels is raw data. We do not support compressed data with this application.

The input stream follows the CTN Simple Object Specification described in Appendix A of this document.

Options

-b	Use Big-Endian order to store the object (not Part 10)
-p	Read raw pixel data from the file *pixels*.
-t	Files are processed in DICOM Part 10 format
-v	Verbose mode; place the DCM facility in verbose mode
filein	File that contains original DICOM object.
fileout	File that stores the modified DICOM object.

Notes

dcm_modify_object is a similar application with more features.

8.10 dcm_modify_object

Name

dcm_modify_object - modify data elements in an existing DICOM information object

Platforms

Unix, Windows

Synopsis

```
dcm_modify_object [-b] [-i inputfile] [-p pixels] [-t] [-v] [-x]
     filein fileout
```

Description

dcm_modify_object reads an existing DICOM information object from a file and modifies individual data elements based on input read from stdin or the file specified with the –i switch. This object contains only those attributes specified by the user; there is no guarantee by this program that the output is a "legal" DICOM composite object, a DICOM normalized object, or any other type of DICOM object. The tool merely provides a translation mechanism.

The user may also modify pixel data with the –p switch. The file format of pixels is raw data. We do not support compressed data with this application.

The input stream follows the CTN Advanced Object Specification described in Appendix A of this document.

Options

-b	Use Big-Endian order to store the object (not Part 10)
-i	Use input file for the ASCII modification instead of standard input.
-p	Read raw pixel data from the file *pixels*.
-t	Files are processed in DICOM Part 10 format
-v	Verbose mode; place the DCM facility in verbose mode
-x	Only process pixel data; there are no other modifications to make
filein	File that contains original DICOM object.
fileout	File that stores the modified DICOM object.

Notes

dcm_modify_elements is a similar application with a simpler input syntax.

8.11 dcm_print_dictionary

Name

dcm_print_dictionary - Print the DICOM data dictionary

Platforms

Unix, Windows

Synopsis

dcm_print_dictionary

Description

dcm_print_dictionary prints the entries in the DICOM data dictionary that are maintained by the DCM facility. Each group is printed (with group number and a brief title) followed by the entries in each group. Output for each attribute includes the tag, value representation and attribute name.

Notes

The DCM facility does not maintain value multiplicity, so we make no effort to print that.

8.12 dcm_resize

Name

dcm_resize - resize a DICOM image

Platforms

Unix, Windows

Synopsis

dcm_resize [-b] [-c cols] [-r rows] [-v] input output

Description

dcm_resize resizes an existing DICOM image by applying pixel averaging. The user specifies the new size of the image by specifying a different number of rows and/or columns through command line switches.

The following options are available:

-b	Read data from input files assuming big-endian format.
-c cols	Produce new image with cols columns.
-r rows	Produce new image with rows rows.
-v	Verbose mode. Turn on verbose mode for the DCM facility.

Notes

This will only work for monochrome images.

8.13 dcm_rm_element

Name

dcm_rm_element - remove one data element from a DICOM information object

Platforms

Unix, Windows

Synopsis

dcm_rm_element [-b] [-v] group element filein fileout

Description

dcm_rm_element reads the DICOM information object stored in filein, removes the element specified by (group, element) and stores the result in fileout.

The following options are available:

-b	Read data from input file assuming big-endian format.
-v	Verbose mode. Turn on verbose mode for the DCM facility.

8.14 dcm_rm_group

Name

dcm_rm_group - remove a group from a DICOM information object

Platforms

Unix, Windows

Synopsis

dcm_rm_group [-b] [-v] filein fileout group [group ...]

Description

dcm_rm_group reads the DICOM information object stored in filein, removes one or more groups as specified by group, [group ...] and writes the result in fileout.

The following options are available:

-b	Read data from input file assuming big-endian format.
-v	Verbose mode. Turn on verbose mode for the DCM facility.

8.15 dcm_template

Name

dcm_template - dump the description of the required Information Entities, Modules and Attributes to the standard output.

Platforms

Unix

Synopsis

dcm_template [-v] SOPClass [SOPClass]

Description

dcm_template uses functions in the IE facility to dump the required Information Entities, Modules and Attributes for an Information Object in the SOP Class. Only the english description of the Information Entity, Module and Attribute will be printed.

The following option is available:

-v	Verbose mode. Turn on verbose mode for the DCM facility.

See Also

Programmer's Guide to the DCM Facility

Programmer's Guide to the IE Facility

DICOM V3, Part 3

8.16 dcm_verify

Name

dcm_verify - dump the contents of a DICOM V3 file, the required Information Entities, Modules and missing Attributes (if any) to the standard output.

Platforms

Unix, Windows

Synopsis

dcm_verify [-b] [-t] [-v] filename

Description

dcm_verify uses functions in the IE facility to examine an Information Object and determine which Information Entities, Modules and Attributes are present and missing (per the rules stated in Part 3 of the DICOM V3 Standard). The information printed for each Information Entity and Module includes:

· Structure type

· Short english description

· Requirement type (mandatory, optional)Status (complete, incomplete or missing)

The information printed for each Attribute includes:

· Tag (group and element number)

· Attribute requirement type

· Short english description

· Value of the data element

The following options are available:

-v	Verbose mode. Turn on verbose mode for the DCM facility.
-b	Read data from input file assuming big-endian format.
-t	Read file with DICOM Part 10 format

See Also

Programmer's Guide to the DCM Facility

Programmer's Guide to the IE Facility

DICOM V3, Part 3

8.17 dcm_x_disp

Name

dcm_x_disp - display a DICOM image file on an X display.

Platforms

Unix

Synopsis

dcm_x_disp [-b] [-t] [-w width][-h height][-W window][-C center] <dicom image file>

Description

An X-based display utility used to view a DICOM image file. This utility uses the same image handling algorithms as ctndisp, making it useful for testing and debugging. The only required parameter is <dicom image file>. The optional parameters are:

-b	use the ORDERBIGENDIAN option when retrieving image data.
-t	Image file is in DICOM Part 10 format.
-w	make the width of the display "width" pixels instead of a full screen default.
-h	make the height of the display "height" pixels instead of a full screen default.
-W	force the window for this image to be "window".
-C	force the center or level for this image to be "level".

Notes

dcm_x_disp prints various statistics about the image to standard output after displaying the image data. In addition, dicom_x_disp requires certain attributes to be present before an image can be successfully displayed. These required attributes are:

0028,0002

Samples per Pixel

0028,0101

Bits Stored

0028,0010

Rows

0028,0102

High Bit

0028,0011

Columns

0028,0103

Pixel Representation

0028,0100

Bits Allocated

7FE0,0010

Pixel Data

The following optional parameters will be processed if present:

Photo Interpretation (0028,0004) (MONOCHROME1 or MONOCHROME2 only)

Window Center (0028,1050)

Window Width (0028,1051)

Rescale Slope (0028,1052)

Rescale Intercept (0028,1053)

8.18 dicom_echo

Name

dicom_echo - create an Association with a server and send one or more C-ECHO requests

Platforms

Unix, Windows

Synopsis

dicom_echo [options] node port

Description

dicom_echo requests an Association with a server running on node node and TCP/IP port address port. dicom_echo uses a default set of called and calling AE titles, but allows the user to override these with command line arguments. The default mode for the program is to establish an Association and send one C-ECHO request. The caller can request a number of C-ECHO requests be sent by using the -r switch.

Options:

-a title	Use title as the Calling Title in the Association Request
-c called	Use called as the Called Title in the Association Request
-d	Drop association after echo requests
-m mode	Mode for SCU/SCP negotiation (SCU, SCP, SCUSCP)
-n num	Number of network connections requested
-p	Dump service parameters after association request
-r repeat	Send repeat C-ECHO requests (per network/association request)
-v	Place facilities in verbose mode for debug purposes.
-x	Do not release associations when finished with echo

Notes

A number of other (strange) switches have made their way into this application. These are normally used as a debugging tool to force dicom_echo to do something out of the ordinary so we can observe how the receiving application performs.

8.19 gqinitq

Name

gqinitq - initialize a new queue.

Platforms

Unix

Synopsis

gqinitq <qid> <queue size> <element size>

Description

gqinitq creates a new queue with id <qid>, with <queue size> number of elements, where each element is <element size> bytes.

Notes

The environment variable QUEUE_DIRECTORY must be set before using this utility.

See Also

All other gq utilities and the GQ Facility.

8.20 gqkillq

Name

gqkillq- remove an existing queue from the system.

Platforms

Unix

Synopsis

gqkillq <qid> <element size>

Description

gqkillq removes the queue from the system whose identifier is <qid> with an element size of <element size>. The element size check is a safety precaution.

See Also

All other gq utilities and the GQ Facility.

8.21 kill_ctndisp

Name

kill_ctndisp - remove ctndisp from the system

Platforms

Unix

Synopsis

kill_ctndisp <queue_number>

Description

kill_ctndisp kills the ctndisp using <queue_number> from the system. In addition, it removes the queue identified by <queue_number> from the system. Any resources (semaphores and shared memory) used by ctndisp are returned to the system.

Notes

Processes still using the queue specified by <queue_number> may behave unexpectedly after *kill_ctndisp* is used.

See Also

ctndisp, enq_ctndisp, pq_ctndisp

8.22 object_viewer

Name

object_viewer - a Motif application for navigating DICOM information (image) objects

Platforms

Unix

Synopsis

object_viewer

Description

object_viewer is a simple Motif-based application that allows a user to examine a DICOM image information object. The user can open a file and see the attributes present in the file. Data is broken down according to the groupings that are suggested in Part 3 of the DICOM standard. This means the program presents Information Entities, Modules and Attributes to the user.

Notes

This programs uses the IE facility to break down image information objects. Not all objects are implemented.

See Also

Programmer's Guide to the IE Facility

8.23 pdus

Name

ex1_initiator

ex2_initiator

ex3_initiator *ex3_acceptor*

ex4_initiator *ex4_acceptor*

Platforms

Unix

Synopsis

ex1_initiator [-c calledTitle] [-v] node port

ex2_initiator [-c callingTitle] [-d fac] [-m maxPDU] [-t calledTitle] [-v] node port

ex3_initiator [-c callingTitle] [-d fac] [-m maxPDU] [-t calledTitle] [-v] node port

ex4_initiator [-c callingTitle] [-d fac] [-m maxPDU] [-t calledTitle] [-v] node port

ex3_acceptor [-f] [-p] [-v] port

ex4_acceptor [-a] [-d fac] [-f] [-m maxPDU] [-p] -v

Description

These applications are used to generate assocation request and accept messages and to dump the protocol data units (PDUs) used in those messages in hexadecimal (ASCII) format. These PDU examples correspond to the the examples which are detailed in the document DICOM Network Examples: Example Protocol Data Units. The applications are used in pairs. *ex1_initiator* and *ex2_initiator* initiate associations and can be used with *simple_storage*. *ex3_initiator* should be used with *ex3_acceptor; ex4_initiator* should be used with *ex4_acceptor*.

Options:

-a callingTitle	Abort association during conversation (debugging tool).
-c	Set the calling AE title to something other than default value.
-d fac	Place facilility fac in verbose mode.
-f	Forgiving mode. Allow associations with some incorrect parameters in request.
-m maxPDU	Set the size of the maximum length of the PDV list in a P-DATA PDU.
-p	Dump association parameters in a human-readable format.
-t calledTitle	Set the called AE title to something other than the default value.

-v	Place DCM, DUL and SRV facilities in verbose mode.

The applications generate Association requests which demonstrate the following:

Example 1 Multiple presentation contexts with

Example 2 A single presentation context with multipler

Example 3 One SOP Class repeated several presentation contexts with a different transfer syntax

Example 4 The Detached Patient Management Meta SOP Class showing SCU/SCP role negotiation.

Notes

You will need to use the -f switch on the server applications and the -v switch on clients and servers to force the applications to dump the PDUs to the stderr.

8.24 print_client

Name

print_client - test print servers by establishing an Association and sending print commands.

Platforms

Unix

Synopsis

print_client [-c calledAETitle] [-f films] [-i imageFormat] [-p] [-s] [-t callingAETitle] [-v] node port file...

Description

print_client is designed as a test program for print servers. It establshes an Association with a print server and implements a basic film print session. The user specifies optional arguments to control the session.

Options:

-c calledAETitle	Use calledAPTitle in the Association request. This should be the AE Title of the print server.
-f films	Specify number of films to be printed. Default is 1.
-i imageFormat	Parameter to be specified for Image Display Format attribute in Basic Film Box Presentation Module (refer to Part 3, Table C.13.3-1, July, 1993). For example: STANDARD\2\2.
-p	Dump Association parameters after Association established (for debugging).
-s	Silent mode; do not print results of all print commands.
-t callingAETitle	Use callingAETitle in the Association Request as the AE Title of the client.
-v	Use verbose mode for DUL and SRV facilities.
nod	The host name that is running a print server.
port	The TCP/IP port number of the print server.
file	The name of one or more files that contain preformatted images to be printed. Each file is opened and used as the preformatted image in the Basic Image Box (set via the N-SET command).

8.25 send_image

Name

send_image - send one or more images to a receiver application by using the DICOM Storage Class.

Availability

Unix, Windows

Synopsis

send_image [-a application] [-c called] [-m maxPDU] [-p] [-q] [-r] [-s SOPName] [-t] [-x FAC] [-v] node port image [image ...]

Description

send_image is an application that acts as a DICOM Storage SCU for a number of different SOP classes. It establishes one or more DICOM associations with an application that acts as a Storage SCP and sends one or more images using the DICOM Storage SOP Class (C-STORE command). The user supplies a list of one or more images to send to the receiving application. send_image proposes one storage class per association. If it finds an image in the list of a different SOP class than the prior image, send_image closes the current association and initiates a new one.

Options:

-a application	Set the calling (local) AE title to application. Default is DICOM_TEST.
-c called	Set the called (remote) AE title to called. Default is DICOM_STORAGE.
-m maxPDU	Set my maximum PDU to maxPDU. Default is 16384.
-p	Alter image by sending minimal pixel data.
-q	Place application in quiet mode. Suppresses some messages.
-r	Check the status in the C-Store response message. If status is not success, then stop sending images. Normally, send_image just keeps trying to send images.
-s	SOPName Force send_image to create an association with one SOP class before reading the image in the list. We use the modality abbreviations for SOPName (CR, CT, MR).
-t	Time the image transfer. Print elapsed time and transfer rate.
-x FAC	Place one facility in verbose mode. Values for FAC are DUL, DCM, SRV.
-v	Place DUL and SRV facilities in verbose mode.

node	Name of the host to connect to.
port	TCP/IP port number of remote application.
image	A list of one or more image files to be sent.

The default file format is CTN format. If send_image fails to open the file as a CTN file, it will try as a DICOM part 10 file.

Notes

The -p switch is used for tests we run when we really only care about the header information and not the pixels. We replace the pixels in the original image with 4 bytes of pixel data. These "smaller" images are useful for testing database operations.

See Also

simple_storage

8.26 simple_storage

Name

simple_storage - a storage application that implements some of the storage classes and some of the query/retrieve classes (as an SCP)

Platforms

Unix, Windows

Synopsis

simple_storage [-a] [-c title] [-d fac] [-i] [-m max] [-n naming] [-p] [-s]
[-t trips] [-v] [-w] [-z sec] port

Description

simple_storage implements several DICOM storage SOP classes and the DICOM verification SOP class. It communicates with clients that wish to use some of the query/retireve SOP classes, but it always gives the same response and moves the same images in response to a move request, so it is not very interesting. It is a useful tool for checking the basic steps of Association establishment and image storage.

Options:

-a	Abort the Association after receiving one image (debugging tool)
-c title	Set the AE title of this application. If set, all client applications must use the proper "Called Application Entity Title" in their association requests.
-d fac	Place fac (DCM, DUL, SRV) in verbose mode.
-i	Ignore some incorrect parameters (like called application title) in Association request.
-m max	Set the maximum received PDU in the Association reply to max (default 16384)
-n naming	Using convention in file naming to create directory/file names.
-p	Dump the Association request parameters to stdout.
-s	Silent mode. Don't dump attributes of received images.
-t trips	Execute the main loop trips times (debugging tool).
-v	Place DUL and SRV facilities in verbose mode.
-x dir	Set the base directory for creating images. Default is current directory.
-w	Wait flag. Wait for 1 second in callback routines (debugging).
-z sec	Wait for sec seconds before releasing association

port	TCP/IP port number used to accept incoming network connections

Notes

simple_storage no longer assumes a specific AE title. Clients that connect with the application may use an arbitrary “Called” Application Entity Title. If you invoke the program with the –c switch, you override that behavior and require client applications to propose an association with the proper called AE title.
simple_storage is a single threaded application. It supports multiple associations, but sequentially.
simple_storage creates files using one of two naming conventions. The default convention is <modality>/<SOP Instance UID>. For example:

```
CT/1.2.840.12345.123.456.789
```

The program allows the user to specify a naming convention using the -n switch. The naming convention file is a series of lines which describe how directories and files are created. Each line in the file describes one level of directory or file creation. You can have multiple directory specifications but only a single file specification. These directives are supported:

```
A       create a directory with calling AE title
C       create a directory with called AE title
D gggg eeee   DEFAULT
        create a directory with the string taken from (gggg, eeee).
        If the attribute is empty or missing, using the value DEFAULT
        for the directory name.
F gggg eeee DEFAULT
        create a file with the string taken from (gggg, eeee).
        If the attribute is empty or missing, use the value DEFAULT
        for the file name.
```

The syntax also supports lines that begin with `#' as comments. The characters `^' and ` ` are mapped to `_'.
Here is a common specification:

```
A
D 0008 0050 ACCESSION
D 0020 000E SERIESINSTANCE
F 0008 0018 SOPINSTANCE
```

The attributes (0020, 000E) and (0008, 0018) are DICOM type 1 attributes. Accession number (0008, 0050) is a type 2 attribute and may be empty. In this case, we would have a directory named "ACCESSION" with multiple series in it for images that arrive with no value in that attribute.
I would like to use image number (0020, 0013) for the file specification, but some vendors don't provide good values in that attribute.

MIR CTN Scripts

This section describes scripts that were created for configuring the demonstration or running the demonstration. These scripts assume (or create) a directory structure that has a single ROOT directory and subdirectories for the various parts of the demonstration. The default ROOT directory used by MIR is /mir_ctn. You can use this root (create a real directory or a soft link) or you can choose your own root directory. The scripts that create directories or database files ask for the root directory. The scripts that start programs have the root directory defined at the top and do not prompt you for them.

You can use these scripts to setup the CTN demonstration in your lab and can use our default root. There is no requirement that you do so.

Print Scripts

create_icons

This is the top level script for creating the files needed for the print_mgr application. It invokes a separate script (icon_script) for each study that is included in the print demonstration. It uses the set of common images found in ROOT/img/db/common. It creates print files and icons in ROOT/print/images and ROOT/print/db. This script assumes that specific subdirectories exist in ROOT/img/db/common and ROOT/print/images for the common images. You should have copied the set of common images to ROOT/img/db/common. print_layout will create the subdirectories needed for the print demonstration.

icon_script

This script is used to populate a print database and an icon database with the information that is used by print_mgr. It is called by create_icons. We do not assume that you will run it.

print_layout

This script is used to create a layout for the print demonstration. It assumes a default ROOT directory of /mir_ctn and prompts the user to allow the user to change that assumption. It creates these subdirectories of ROOT: print, print/config, print/db, print/images. One subdirectory is created in print/images for each study in the set of common images (cr1, cr2, cr3, ct1, ct2, ...). This script does not copy a configuration file into print/config nor does it create the print database and icon files needed by print_mgr.

start_print_client

This script defines the ROOT directory and environment variables and starts the print_client program. The only argument to this script is the vendor name

9 MESA Utilities

9.1 cfind

Name
cfind – send a DICOM C-Find request to a server application and print results
Platforms
Unix, Windows
Synopsis
cfind [-a title] [-c title] [-d delta] [-f file] [-o output] [-p] [-v] [-x class] node port
Description
cfind sends a DICOM C-Find query to a DICOM SCP of one of the query classes. This has been tested with query/retrieve and MWL SCPs. Other classes that use the same query fundamentals could be incorporated.
The user specifies a DICOM query using one of two mechanisms.
If the user specifies the –f switch, the *file* argument is the name of a file in CTN format (not Part 10) that contains the exact query to send to the SCP. This includes both the specification for query and return keys and may include sequences. An addition to this mode is the –d switch that enables the user to specify a text file with modifications to the query. The format of the text file follows the MESA Delta Object Specification specified in Appendix A of this document. The typical use of the –f and –d switches is to form a baseline query with the –f switch (say the return keys) and add specialization with the –d switch (say the query keys).
If the user omits the –f switch, the query object is read from text supplied in the standard input. The format of the text that describes the query object is CTN Simple Object Specification found in Appendix A. The –d switch (for delta file) can also be applied to the query object after it has been read from the standard input.
All responses are dumped in ASCII format as they are received. If the user specifies the –o switch, responses are also stored in files (CTN format) in the directory specified by the –o argument (*output*).
The default SOP class for this application is the Modality Worklist Information Model – FIND. To specify a different SOP class, use the –x switch to give the SOP class UID or an abbreviation. Use the –p switch to list the abbreviations. For example:

```
-x STUDY
```

will specify the SOP class: Study Root Query/Retrieve Information Model – FIND.
Options

-a	Specify calling Application Entity Title (this application)
-c	Specify called Application Entity Title (server)
-d	Name a delta file to apply to query after it has been entered
-f	Name CTN formatted file with query specification
-o	Specify directory to store query output (one file/response)
-p	List the abbreviations for SOP classes and exit

-x	Specify the SOP class to use for the query. Default is MWL.
-v	Verbose mode. Print the parameters for Association negotiation.
node	Host name or IP address of system to contact
port	TCP/IP port number of server application

Notes

9.2 cfind_evaluate

Name
cfind_evaluate – evaluate a C-Find query for “correctness”
Platforms
Unix, Windows
Synopsis
cfind_evaluate [-v] test file [file…]
Description
cfind_evaluate examines one or more C-Find requests and evaluates them for adherence to the DICOM Standard and IHE Technical Framework. If the evaluation is successful, the application returns 0. The application returns 1 when the evaluation is not successful.
The parameter *test* names the test to be run. The values for *test* are:

PATIENT	Evaluate Patient Root model query
STUDY	Evaluate Study Root model query
MWL	Evaluate MWL query
IMG_AVAIL	Evaluate Image Availability query

Following are some of the tests
PATIENT
Unimplemented.
STUDY
For these queries, two types of tests are run. The first test is to see if the user has only specified the unique matching key at the query level above the current level. For example, for a series level query, the only key at the study level should be the Study Instance UID. The second test is to determine if the query contains keys for levels that are lower than the current level. Again, for a series level query, no keys at the image/SOP instance level should be included.
MWL
Unimplemented.
IMG_AVAIL
Image availability queries are based on the premise that the client has received MPPS information that includes image references. Queries should only be made at the image level to determine if they are present. This test evaluates queries and determines if the user has specified queries above the image level.
Notes
Because this evaluation is based on our interpretation of those documents, users may find this evaluation subjective. Based on our experience, groups that develop C-Find queries sometimes include attributes that are not part of the query model. Some applications ignore these extra attributes while other attributes get upset and do not return a response. The purpose of this application is to make the queries “safer” so that the client is not surprised by responses from one server and the lack of responses from a different server.
Drawing the line between where an Image Availability query stops and where a new query from

a client about study or series information begins is difficult. Users of the application should understand the intent of the query and perform the proper evaluation. That is difficult in the test environment where these tools are most often use.

9.3 cfind_image_avail

Name
cfind_image_avail – send an Image Availability C-Find request to a DICOM query SCP
Platforms
Unix, Windows
Synopsis
cfind_image_avail [-a title] [-c title] [-o output] [-v] node port <MPPS status file>
Description
cfind_image_avail sends an Image Availability query to a DICOM query SCP. This query is based on the image references found in the MPPS status file. The status file is created by MESA Order Fillers or Image Managers as a result of MPPS N-Create and N-Set messages. This file contains the final value of all attributes as defined by the application of those messages. This is not a standard DICOM object, but is the sum of the attributes created/set by those MPPS operations. The application reads that file, extracts the Study Instance UID and Series Instance UID, and sends an Image level query where the SOP Class UID (0008 0016) and SOP Instance UID (0008 0018) are left as return keys (0-length).
Each C-Find response is dumped in ASCII to the standard output. If the user specifies the –o switch, each response is stored in a (CTN formatted) DICOM file in the *output* directory.
Options

-a	Specify calling Application Entity Title (this application)
-c	Specify called Application Entity Title (server)
-o	Specify directory to store query output (one file/response)
-v	Verbose mode. Print the parameters for Association negotiation.
node	Host name or IP address of system to contact
port	TCP/IP port number of server application

Notes
Maybe we should add other options to allow the user to specify SOP Instance UIDs in the query.

9.4 cfind_mwl_evaluate

Name

cfind_mwl_evaluate – evaluate a single MWL response from an SCP

Platforms

Unix, Windows

Synopsis

cfind_mwl_evaluate –v <test file> <master file>

Description

cfind_mwl_evaluate examines a single response to an MWL query and compares that response to the "gold standard" found in the *master file*. The application compares attributes that can be set as part of a test package (code values, patient ID, patient name). Other attributes are required to be present but are under control of the MWL system (such as Study Instance UID). For these attributes, the test determines that the value is present.

The application returns 0 if the evaluation is successful. The application returns 1 if one or more tests fail.

Options

-v	Verbose mode
test file	The name of the file under test
master file	The MWL response that is considered truth

Notes

The user should have already determined by some other mechanism that the response under test should be matched with the master file. For example, you might query the worklist system for a specific patient knowing that only one procedure should be scheduled. You might query for all procedures, and then search the responses for the one procedure with the code value specified in some test plan.

9.5 cfind_resp_evaluate

Name
cfind_resp_evaluate – evaluate a set of C-Find responses
Platforms
Unix, Windows
Synopsis
cfind_resp_evaluate [-s gggg eeee] [-v] <tag> <mask> <test dir> <master dir>
Description
cfind_resp_evaluate is used to evaluate a set of responses provided by a DICOM query SCP. These can be MWL or Patient or Study Root model query/responses. The user queries a system under test (using cfind) and places all of the results in a directory (*test dir*). The user should send the same query to the master system using cfind and place those results in a separate directory (*master dir*). This application will then search both directories, correlate the responses, and evaluate corresponding responses.
The user specifies one attribute to be used as the primary key to correlate the responses in the master and test directories. The *tag* argument is of the form `gggg eeee` and specifies that attribute. If the attribute is part of a sequence, the –s switch can be used to specify the sequence. The value in *tag* will define the attribute in the sequence to be used as the primary key.
The mask argument specifies a text file used to determine how to evaluate attributes in the response. Each line of the text file designates one attribute to be evaluated. The format of the line is one of the following:

```
gggg eeee c
GGGG EEEE gggg eeee c
```

The value gggg eeee specifies the attribute to be tested. If the line is of the second form, the GGGG EEEE value indicates a sequence, and the gggg eeee indicates the attribute within the sequence.
The character 'c' is one of the following:

=	Compare the attribute for equality. Attribute must match exactly.
O	Optional attribute. Test to see if attribute is present. Do not test value. Do not fail if attribute is not present.
P	Test to see if attribute is present and length is not zero. Fails if zero length or not present.

Blank text lines or lines that begin with '#' are ignored.
The application returns 0 if all responses evaluate successfully and 1 if one or more responses fail.
Notes
This application expects the number of responses to be the same in the test and master directories. If the user cannot specify an attribute that will uniquely distinguish the responses, this application will not work properly.

9.6 cfind_study_probe

Name
cfind_study_probe – send a series of queries from the STUDY to IMAGE level for the Study Root Information model.
Platforms
Unix, Windows
Synopsis
cfind_study_probe [-a title] [-c title] [-o output] [-v] node port <patient ID> q1 q2 q3
Description
cfind_study_probe is used to query a system that supports the DICOM Study Root Information Model. The application sends queries at the STUDY, SERIES and IMAGE levels to the system. That is, we follow the trail of responses through each level of the hierarchy. The user selects a particular patient for query by specifying the *patient ID* argument.
As each response is processed, it is dumped in ASCII format to the standard output. If the user specified the –o switch, the responses are also stored in CTN format in the *output* directory.
The *q1*, *q2* and *q3* arguments name files stored in CTN format that contain the queries for the STUDY, SERIES and IMAGE levels.
Options

-a	Specify calling Application Entity Title (this application)
-c	Specify called Application Entity Title (server)
-o	Specify directory to store query output (one file/response)
-v	Verbose mode. Print the parameters for Association negotiation.
node	Host name or IP address of system to contact
port	TCP/IP port number of server application
patient ID	Identifies the patient to select for first query
q1	The return/query keys for STUDY level query
q2	The return/query keys for SERIES level query
q3	The return/query keys for IMAGE level query

Notes
The user must construct the query files to contain the unique keys (as return attributes) for each query level. This places an extra burden on the user, but also allows the user to modify those queries and use the attributes as query keys as well. For example, if you knew a Series Instance UID, you could program that value into q2.
You could use that same mechanism to have a combination of query and return keys in the query files. For example, you could include Modality (0008 0060) as a query key at the SERIES level and only query for MR images. You could include SOP Class UID at the IMAGE level and only query for certain SR SOP instances.

9.7 cmove

Name
cmove – send a DICOM C-Move request to a server application.
Platforms
Unix, Windows
Synopsis
cmove [-a title] [-c title] [-d delta] [-f file] [-p] [-v] node port dest
Description
cmove is the cousin of the *cfind* application. It sends a DICOM C-Move request to a server and processes the responses. Please read the cfind documentation for a description of how to construct the query to be sent to the server.
The query that is sent needs to include the AE title of the destination. Note that this application does not include a DICOM storage SCP. The user is responsible to run a separate application if the images are to be returned to the user. In many cases, this application is used to direct the images to a third party DICOM Storage SCP.
Options

-a	Application title of this application
-c	Called AP title to use during Association setup
-d	Text file containing delta to apply to C-Move request object
-f	File containing a dcm object with query
-p	Dump service parameters after Association Request
-v	Verbose mode for DUL/SRV facilities

node	Node name of server
port	Port number of server
dest	Destination AE title for move

Notes

9.8 cmove_study

Name

cmove_study – send STUDY level queries and C-Move requests to a DICOM query/retrieve SCP

Platforms

Unix, Windows

Synopsis

cmove_study [-a title] [-c title] [-o] [-v] [-Z] node port tag value dest

Description

cmove_study is a DICOM C-Move application that is different from the cmove program. This application only supports the Study Root Information Model. More importantly, the application has a different operational model. The user specifies one attribute (tag value) to be used as a query key in a STUDY level C-Find request. The application builds the C-Find request and adds Study Instance UID (0020 000D) as a return key (zero-length attribute). This C-Find request is sent to the query/retrieve SCP.

The *cmove_study* application processes each C-Move response sequentially. Each response should yield one Study Instance UID. That value is used as the unique key in a STUDY level C-Move request to the same server. The C-Move request is formatted to include the destination AE Title taken from the *dest* argument.

The best examples for using this program would be to request all of the studies for a patient with Patient ID AB (0010 0020 AB) or for a study with Accession Number XYZ (0008 0050 XYZ). As with the *cmove* program, this application does not include a DICOM Storage SCP.

Options

-a	Specify calling Application Entity Title (this application)
-c	Specify called Application Entity Title (server)
-o	Specify directory to store query output (one file/response)
-v	Place DUL and SRV facilities in verbose mode. Dump association parameters
-Z	Use verbose mode for each C-Move response.
node	Host name or IP address of system to contact
port	TCP/IP port number of server application
tag	DICOM tag of attribute to use as query key (gggg eeee)
value	Value of query key attribute
dest	AE title of C-Move destination

Notes

compare_dcm

Name

compare_dcm – compare two DICOM objects, one a gold standard, the other a test file

Platforms

Unix, Windows

Synopsis

compare_dcm [-g] [-m maskfile] [-o] [-t] [-v] [-z] master test

Description

compare_dcm compares a test DICOM object to a “gold standard” object. These objects can be composite object (images, SR objects) or normalized objects (MPPS objects).

The arguments *master* and *test* name files stored in CTN format. The application walks through the attributes in the objects in numeric order. In the absence of a mask file, each attribute is compared to the corresponding attribute in the peer object. If the attribute is missing or has a different value, a difference is noted. The application returns 0 if no differences are noted and 1 if there are any differences.

The user can restrict the attributes that are tested by specifying a mask file with the –m switch. This file lists the attributes that should be examined. Each line lists a single attribute in the format: gggg eeee. Attributes not listed in the mask file are ignored.

Options

-g	Remove group length attributes before processing
-m	Specify name of a mask file that names the attributes to evaluate
-o	Verbose mode (deprecated)
-t	Input files are DICOM Part 10 files
-v	Verbose mode
-z	Perform some format conversions when opening files. Change old date/time formats to DICOM specification
master	The name of the file that contains the “gold standard”
test	The name of the file that is to be evaluate

Notes

Comparing DICOM objects without specialization for SOP classes is a difficult task. In fact, a separate application that considers the SOP class should be written. This application is most useful in the MESA environment for testing attributes that can be specified in a test plan and are independent of the SOP class. For example, you can use DICOM MWL to specify attributes such as Study Instance UID, Patient ID and certain code values. This application can be used to test for the proper values of those attributes.

9.9 compare_hl7

Name
compare_hl7 – compare two HL7 messages, one a gold standard, the other a test file
Platforms
Unix, Windows
Synopsis
compare_hl7 [-b base] [-d definition] [-m mask] [-v] master test
Description
compare_hl7 compares a test HL7 message against a master message. The messages are found in the files named by the *test* and *master* arguments to this application. In the absence of a mask file, the application walks through all of the segments and fields of the messages and compares them for equality.
A mask file (specified by the –m switch) allows the application to selectively compare segments and fields. The mask file is an ASCII file. Text lines contain a segment name in [] brackets (for example: [ORC] or the number of a field within a segment (for example 3). To compare all fields within a segment, only list that segment. To compare a subset of fields within a segment, list the segment and follow with the field numbers. In the example below, we test 3 fields in the ORC segment and all of the OBR segment.

```
[ORC]
1
3
12
[OBR]
```

Lines that are empty or begin with '#' are ignored.
The application returns 0 if all fields match and 1 if there are any differences.
Options

-b	Set base for HL7 definitions (default is $MESA_TARET/runtime).
-d	Set extension for HL7 definitions (default is ihe)
-m	Set mask file to determine attributes to compare
-v	Verbose mode
master	The file considered the gold standard
test	The file under test

Notes
Comparing an entire HL7 message or an entire segment is a difficult task. Optional segments or fields, or fields whose values cannot be anticipated make the comparison difficult. This program is best used with a mask file. See also *hl7_evaluate* to evaluate the format of the HL7 message.

9.10 cstore

Name
cstore – store one or more composite objects to a DICOM Storage SCP
Platforms
Unix, Windows
Synopsis
cstore [-a title] [-c title] [-d delta] [-l level] [-n] [-p] [-v] node port file [file…]
Description
Cstore is the MESA younger brother of the original CTN send_image program. It provides the same basic function (DICOM C-Store command for composite objects), but has some different features.
The *file* argument or arguments list files or directories to search. If *file* names a directory, the application will attempt to send every file in that directory. The application normally recurses through the directory hierarchy. The –n switch directs the program to only send the files from the top level directory.
The –d switch that enables the user to specify a text file with modifications to the composite objects. The format of the text file follows the MESA Delta Object Specification specified in Appendix A of this document
Log messages generated by the MESA library are controlled through the –l switch. The default is that no messages are output (to the standard output). The user can enable more logging by setting the level to 1, 2, 3 or 4 (Error, Warning, Conversation, Verbose).
Options

-a	Specify calling Application Entity Title (this application)
-c	Specify called Application Entity Title (server)
-d	Name a delta file to apply to query after it has been entered
-l	Change log level from 0 (no logging). Use 1, 2, 3, or 4.
-n	If one or more files is a directory, do not recurse down
-v	Verbose mode for DUL/SRV facilities. This is independent of –l switch.
node	Host name or IP address of system to contact
port	TCP/IP port number of server application
file	One or more files or directories to store.

Notes

9.11 dcm_print_element

Name
dcm_print_element – print the value of one DICOM attribute
Platforms
Unix, Windows
Synopsis
dcm_print_element [-i index] [-s g1 e1] g2 e2 file
Description
dcm_print_element prints the value of a DICOM attribute to the standard output. This is useful for scripts that want to capture the value of that attribute (and don't want to grab the value from the file that would be created by *dcm_dump_element*).
The DICOM object is in the file named *file* (CTN or DICOM Part 10 format). The application opens the file and dumps the attribute with the tag g2 e2. If the user specifies the –s switch, g1 e1 name a sequence, and g2 e2 name the attribute within the sequence. To get an attribute from something other than the first sequence item, use the –i switch.
Options

-I	Specify sequence index (if –s switch is used). Default is 1.
-s	Specify to find an attribute in the sequence with tag g1 e1
g2 e2	Tag of attribute to dump (gggg eeee)
file	Name of the file to open

Notes
We use this a lot in our perl scripts. For example:
```
    $v = `$MESA_TARGET/bin/dcm_print_element 0008 0050 x.dcm`;
```
would extract the Accession Number and place it in the variable $v.

dcm_ref_sop_seq
Name
dcm_ref_sop_seq – not supported
Platforms
Unix, Windows
Synopsis
dcm_ref_sop_seq

Description

Notes

9.12 ds_dcm

Name
ds_dcm – data server, DICOM
Platforms
Unix, Windows
Synopsis
ds_dcm [-c title] [-d FAC] [-f] [-i] [-j] [-l level] [-m maxPDU] [-p] [-t trips] [-v] [-y] [-z sec] config
Description
ds_dcm is a data server for DICOM functions. It is most often used for IHE Image Managers. The application uses a configuration file to define the supported services and other features. The description of this application is too complex for this document. This application is more fully described in the document *MESA Server Applications*.
Options

-c	Set AE title of this application. Default is that the application does not care
-d	Place one of the CTN facilities in verbose mode. FAC is one of DCM, DUL, SRV
-f	Use the fork model; fork a copy of the application for every association
-I	Forgive mode; forgive some errors in asscociation requests (such as AE title)
-j	Use thread model; start a new thread for each association
-l	Set the log level (0-4, default is 0: none)
-m	Set the maximum received PDU in the Association RP to *maxPDU*
-p	Dump the Association RQ parameters
-t	Execuate the main loop *trips* times (debugging tool)
-v	Place DUL and SRV facilities in verbose mode
-y	Print connection stats for each trip

Notes

9.13 evaluate_storage_commit

Name
evaluate_storage_commit – evaluate the storage commitment response of a DICOM Storage Commitment SCP
Platforms
Unix, Windows
Synopsis
evaluate_storage_commit [-v] master test
Description
Evaluate_storage_commit is used to evaluate the Storage Commitment response of a DICOM Storage Commitment SCP. Typically the user follows these steps:
Store composite objects to a test system
Store composite objects to the “gold standard” system
Send storage commitment request to both systems and record results
Compare storage commitment responses
This application evaluates the dataset found in the N-Event report issued by the SCP. It checks the transaction UID and the list of committed images. It can also test for “error” images. For example, you could request images that have not been stored. These should be returned in the response in the appropriate section.
This application returns these values:
Results match
Transaction UIDs do not match
Failure in test for committed images
Failure in test for failed images

Notes

9.14 export_agent

Name
export_agent – the agent program that exports data on behalf of the MESA Web system
Platforms
Unix
Synopsis
export_agent [-d] [-l level] [-s dir] dbname
Description
export_agent is a component of the MESA Web system. It reads jobs in a work queue and exports images using DICOM C-Store commands. As currently implemented, the program runs one cycle and then exits. The program could be altered to run continuously, or wrapped in a script to invoke it on a regular basis.
This description is lacking detail. Users should refer to the document *MESA Web System* for a description of all components and the database format.
Options

-d	Delete each work order as completed; default behavior is to leave db entries
-l	Set log level to one of 0-4; default is Conversation (3)
-s	Set the directory name for log messages; default is $MESA_TARGET/logs
dbname	Name of the database holding the work queue and other information

Notes

9.15 hl7_evaluate

Name
hl7_evaluate – evaluate the format of an HL7 message
Platforms
Unix, Windows
Synopsis
hl7_evaluate [-b base] [-d def] [-m mask] [-v] file
Description
hl7_evaluate is used to evaluate the "format" of an HL7 message. That is, it examines an HL7 message and tests for required fields. It determines if required fields are populated and can also test the format of the data in some fields.
The mask file includes segment names enclosed in brackets, [], followed by field-restriction definitions. The format for definition is:

Fld#.Comp#.Sub$, Format = <format>, PV = <V1:V2:V3>;

Where

Fld# = Field Number, Comp# = Component Number, Sub# = Sub Component Number. These must be separated by periods.
<format> defines the format of the value
<V1:V2:V3> are possible values that the field can take with : as a delimiter

The <format> definition is as follows:

$ - a character following this symbol may repeat itself
X – alphanumeric
Z – printable character
# - numeric only
any other character means a literal. For example, '.' means that there should be a period.

Lines that are blank or begin with '#' are ignored. Following is an example for a mask file that evaluates the MSH segment of a message:

```
# Comment
[MSH]
1, PV = |;          // Field separator, can only be |
2, PV = ^~\&;       // Encoding characters are also fixed
3, Format = $Z;     // Sending Application, printable characters
4, Format = $Z;     // Sending Facility
5, Format = $Z;     // Receiving Application
6, Format = $Z;     // Receiving Facility
#                   No entries for 7, 8
9.1.0, PV = ORM;    // Message Type
9.2.0, PV = O01;    // Message Type
10, Format = $Z;    // Message Control ID
11, PV = P:T:D;     // Processing ID, can be P, T or D
12, Format = 2.3.1  // We only accept version 2.3.1 messages
```

This program returns the following values:
Successful evaluation
Unable to read the input file
Evaluation failed

Options

-b	Set base directory for HL7 parsing rules; default is $MESA_TARGET/runtime
-d	Set suffix for HL7 parsing rules; default is ihe
-m	Set the mask file used for evaluation
-v	Verbose mode
file	Name of the file to be evaluated

Notes

The –m switch gives the appearance that this is an optional parameter. It is mandatory and the program will fail without this switch and associated mask file.
The program is used in conjunction with *compare_hl7*. This application is useful for some fields that are always fixed (for example, you could set Processing ID to 'P') and to make sure that fields have the proper format even if you cannot test for a specific value.

9.16 hl7_get_value

Name
hl7_get_value – extract one field from an HL7 message and print to standard output
Platforms
Unix, Windows
Synopsis
hl7_get_value [-b base] [-d def] [-h] -f filename [-i index] segment field component
Description
hl7_get_va lue extracts a field from an HL7 message and prints the value to the standard output. This is useful for scripts that want to capture the value of that attribute.
Options

-b	Set base directory for HL7 parsing rules; default is $MESA_TARGET/runtime
-d	Set suffix for HL7 parsing rules; default is ihe
-f	Name of file containing message (required switch)
-h	Print help message
-i	Optional index denotes which repeated segment
segment	The three character segment name
field	An integer, specifying the field number
component	An integer, specifying the component within a field; can be 0

Notes

9.17 hl7_set_value

Name
hl7_set_value – set the value of one component/field in an HL7 message
Platforms
Unix, Windows
Synopsis
hl7_set_value [-b base] [-d def] [-h] –f file [-i index | -a] segment field component value
Description

Options

-a	Optional, instead of –i index, set for all segments
-b	Set base directory for HL7 parsing rules; default is $MESA_TARGET/runtime
-d	Set suffix for HL7 parsing rules; default is ihe
-f	Name of file containing message (required switch)
-h	Print help message
-i	Optional index denotes which repeated segment
segment	The three character segment name
field	An integer, specifying the field number
component	An integer, specifying the component within a field; can be 0
value	Is the new value to place in the updated message

Notes

hl7_to_txt

Name

hl7_to_txt – convert an HL7 message to text and print the result

Platforms

Unix, Windows

Synopsis

hl7_to_txt [-b base] [-d definition] [-h] file

Description

hl7_to_txt reads an HL7 message stored in file and dumps the message in a simple text format to the standard output. Each segment in the message is listed and then each field is listed with its corresponding number.

Options

-b	Set base directory for HL7 parsing rules; default is $MESA_TARGET/runtime
-d	Set suffix for HL7 parsing rules; default is ihe
-h	Print help message and exit
file	Name of file to dump

Notes

This application would be much better if it also printed out the field names and value types of the fields. In the current implementation, the output of this program can be used as the input to *txt_to_hl7*.

9.18 im_dcmps

Name
im_dcmps – deprecated application
Platforms
Unix
Synopsis
im_dcmps

Description

Notes

9.19 im_hl7ps

Name
im_hl7ps – Image Manager HL7 personality server, receives HL7 messages
Platforms
Unix, Windows
Synopsis
im_hl7ps [-a] [-b base] [-d def] [-l level] [-s dir] [-z db] port
Description
im_hl7ps is a data server for HL7 messages sent to an Image Manager
The description of this application is too complex for this document. This application is more fully described in the document *MESA Server Applications*.
Options

-a	Uses analysis mode, meaning capture and store HL7 messages in files
-b	Set base directory for HL7 parsing rules; default is $MESA_TARGET/runtime
-d	Set suffix for HL7 parsing rules; default is ihe
-l	Set the log level (0-4, default is 0: none)
-s	Set the log directory; default is $MESA_TARGET/logs
-z	Set database name; default is imgmgr
port	TCP/IP port used to accept connections

Notes

9.20 im_sc_agent

Name
im_sc_agent – Image Manager Storage Commitment Agent
Platforms
Unix, Windows
Synopsis
im_sc_agent [-a title] [-c title] host port file
Description
im_sc_agent is the Storage Commitment agent for the MESA Image Manager. The MESA Image Manager stores each Storage Commitment request (the N-Action request) in a file. This application reads the file specified by the user, determines which SOP instances are successfully stored/committed and sends the appropriate N-Event report to the system at *host:port*.
The application returns 0 upon success. A non-zero value could indicate a processing error such as the server did not respond at all or a non-zero status value from the server.
Options

-a	Specify calling Application Entity Title (this application)
-c	Specify called Application Entity Title (server)
host	Host name of system running server
port	TCP/IP port of server application
file	Input file with commitment N-Action request

Notes

9.21 kill_hl7

Name
kill_hl7 – send a MESA defined HL7 message to a MESA server
Platforms
Unix, Windows
Synopsis
kill_hl7 [-b base] [-d def] [-e event] [-i] [-n prefix] [-v] host port
Description
kill_hl7 sends a (non-standard) HL7 message to a MESA server. The purpose of the message is to inform the server to perform some administrative task. Two events are defined:

KIL	Shutdown the server application and exit
RST	Reset the server; remove data files and clear database

Options

-b	Set base directory for HL7 parsing rules; default is $MESA_TARGET/runtime
-d	Set suffix for HL7 parsing rules; default is ihe
-e	Set the event code; default is KIL
-i	Interactive mode for sending messages
-n	Set a prefix for Control Message Ids
-v	Verbose mode
host	Host name or IP address of remote system
port	TCP/IP port number of server application

Notes

9.22 mesa_composite_eval

Name
mesa_composite_eval – evaluate DICOM composite objects
Platforms
Unix, Windows
Synopsis
mesa_composite_eval [-o option] [-p pattern] [-v] file
Description
mesa_composite_eval is used to evaluate DICOM composite objects. It works on modules as defined in DICOM PS 3.3 and is supposed to have some intelligence about the types of variables (type 1, type 2, type 3). The pattern object is used to provide comparison values.
The options give the application a better idea of how to evaluate the composite object. An option (-o) of *mwl* indicates the composite object should have been created in the context of having a DICOM Modality Worklist; that implies certain attributes should have values taken from the MWL. An option of *nouid* directs the application to ignore the Study Instance UID (say for the IHE Group Case).
The application returns 0 if the evaluation completes with no differences and a non-zero value otherwise.
Options

-o	Set an option; one of mwl, nouid
-p	Specify a file to use as a pattern
-v	Verbose mode
file	File to be evaluated

Notes
The modules examined are:

- Patient Module
- General Study Module
- Patient Study Module
- General Series Module
- General Equipment Module
- General Image Module
- Image Pixel Module

9.23 mesa_dump_obj

Name
mesa_dump_obj – not supported
Platforms
Unix, Windows
Synopsis
mesa_dump_obj

Description

Notes

9.24 mesa_sr_eval

Name
mesa_sr_eval –
Platforms
Unix, Windows
Synopsis
mesa_sr_eval [-r requirements] [-o option] [-p pattern] [-t template] file
Description
mesa_sr_eval is used to evaluate DICOM Structured Report (SR) objects. This application focuses on the content items in the SR object and ignores other attributes (such as the Patient Module). There are several different tests that are somewhat independent.
If the user specifies a template (-t switch), the application tests for both the proper value in the Template ID and for adherence to that template itself. As of this version, we can test for any template ID; we only can test for the "format" according to TID 2000.
If the user specifies both requirements (-r) and a pattern (-p), the application evaluates the content items in the test object against the pattern. In this case, the user can add a test for Predecessor Sequence Document by adding –o PREDECESSOR.
Finally, the application always runs some simple tests on the Content Items. In the current implementation, these are tests for the proper value type at the top level and tests for the Continuity of Content attribute (0040 A050).
Options

-r	Specify a text file listing required/optional content items
-o	Specify an option; one of (PREDECESSOR)
-p	Specify a file to use as a pattern
-t	Specify the Template Identifier to test for (e.g. 2000:DCM)
-v	Verbose mode
file	Name of the file to evaluate

Notes
The requirements file is a text file that lists the content items to be tested. Some content items are optional and may appear in the pattern and not the file under test. Each line in the file has this format:

```
Index  Type   Comment
```

Index is a number that helps us keep track of which content item is under test. Start numbering at 1 and increment sequentially.
Type is one letter that tells the application how to treat the content item. Valid codes are:

R	Required, perform level 1 test
O	Optional
F	Required, perform level 2 test

Comment describes the content item under test. It will be printed by the application in the event of errors.

In a level 1 test, content items are tested for equivalence of attributes. In a level 2 test, the content item is evaluated to see if the proper attributes are present. That is, for level 1 tests, we think we know what the values should be; for level 2 tests, we only know the content item should be present and test the "form".

Lines that bergin with # are ignored. Following is an example.

```
# Comment
1    R    Language of Content Item and Descendants
2    O    Observer Type (Person)
3    O    Observer Name
4    R    TEXT: Key Object Description
5    F    IMAGE
```

9.25 mesa_storage

Name
mesa_storage – unsupported
Platforms
Unix, Windows
Synopsis
mesa_storage

Description

Notes

9.26 mod_dcmps

Name
mod_dcmps – modality DICOM personality server, DICOM server for modality systems
Platforms
Unix, Windows
Synopsis
mod_dcmps [-d FAC] [-c title] [-f] [-i] [-j] [-k] [-m maxPDU] [-p] [-s log] [-S storage] [-t trips] [-v] [-x dir] port
Description
mod_dcmps is a data server for modality DICOM functions. That is, this is a server application for a modality that accepts and processes DICOM associations from other systems.
The description of this application is too complex for this document. This application is more fully described in the document *MESA Server Applications*.
Options

-d	Place one facility (DCM, DUL, SRV) in verbose mode
-c	Set the AE title of this server
-f	Fork a new process for each connection
-i	Ignore some incorrect parameters in Association request
-j	Use thread model. Span a new thread for each connection
-m	Set the maximum received PDU in the Assocation RP to *maxPDU*
-p	Dump the Association RQ parameters
-s	Set log directory; default is $MESA_TARGET/logs
-S	Set storage directory; default is $MESA_STORAGE/modality
-t	Execute the main loop *trips* times (debugging tool)
-v	Place DUL and SRV facilities in verbose mode
-y	Print connection statistics at each Association
port	The TCP/IP port number to use

Notes

9.27 mod_generatestudy

Name
mod_generatestudy – generate images of GSPS objects for part of a study
Platforms
Unix, Windows
Synopsis
mod_generatestudy [-a <AE title]> [-c code] [-f file] [-i image] [-m mod] [-n] [-o option] [-p patient] [-r name] [-s code] [-t dir] [-y dir] [-z protocol]
Description
mod_generate study generates the images or GSPS objects for part of a study. It typically reads Modality Worklist responses that are stored in a directory and pulls the response for a single Scheduled Procedure Step. It uses that response to build the attributes for the image or GSPS objects. If no MWL directory is specified, the application produces data for the IHE "Unscheduled Case".
The input directory of images are existing images that serve as the basis for creating the data. In many instances, the modality specified by the user does not match the image type. We leave the image SOP class alone and just change the value in the modality attribute (0008 0060). This is probably not in the spirit of the DICOM Standard, but the purpose of this application is to produce "header" data according to IHE guidelines.
This application is (hopelessly) complex, has two many options and is hard for the authors to understand. It needs to run in a scripted environment and would benefit from a redesign.
Options

-a	Set AE title of station (for the unscheduled case)
-c	Performed code (used as table lookup to get action items)
-d	Discontinued case
-f	Specify a configuration file; format depends on data produced
-i	Find input images in directory: *image*
-h	Print this message
-m	Set modality to *mod*. Default is MR
-n	Do not use procedure codes in PPS information
-o	Define an option; legal values are (GSPS)
-p	Set patient ID
-r	Set patient Name
-s	Set a scheduled Action Item code (can repeat this switch)
-t	Define target directory for output
-y	Directory containing MWL responses
-z	Protocol name

Notes
The options file is used for creating one GSPS object. It contains several values that are stored in the GSPS objects as well as used for selecting the images referenced in the GSPS object. These

values are:
INSTANCE_NUMBER, PRESENTATION_LABEL,
PRESENTATION_DESCRIPTION, PRESENTATION_CREATOR, IMAGE_RANGE
For example:

```
    INSTANCE_NUMBER = 100
    PRESENTATION_LABEL = Procedure_6
    PRESENTATION_DESCRIPTION = Images_for_Procedure_6
    PRESENTATION_CREATOR = CHANNIN^DAVID
    IMAGE_RANGE = 1_10
```

The final parameter indicates the images to select from the input image set.
INSTANCE_NUMBER is the Instance Number for the new GSPS object. The underscores found in the values for PRESENTATION_LABEL, PRESENTATION_DESCRIPTION and PRESENTATION_CREATOR are mapped to space (‘ ‘) before placing in the GSPS object.

9.28 mod_newuids

Name
mod_newuids – unsupported
Platforms
Unix, Windows
Synopsis
mod_newuids

Description

Notes

9.29 mod_weld_mwl

Name
mod_weld_mwl – unsupported
Platforms
Unix, Windows
Synopsis
mod_weld_mwl

Description

Notes

9.30 mpps_app_perf_series

Name
mpps_add_perf_series – unsupported
Platforms
Unix, Windows
Synopsis
mpps_add_perf_series

Description

Notes

9.31 mpps_construct

Name
mpps_construct
Platforms
Unix, Windows
Synopsis
mpps_construct [-d <fac>] [-p <patientID>] [-s <g> <e>] f1 [f2 f3]

-d <fac> Place a CTN facility in verbose mode. Recognized values are DCM.
-p <patientID> Begin constructing an MPPS object for patient with this patient ID. In this case, f1 is the output file and f2, f3 are not specified
-s <g> <e> Add a sequence to the object whose group and element numbers are <g> and <e>, respectively. In this case, f1 is the input file (current contents, f2 contains the sequence to be added and f3 is the target output.
With no switches, combine the attributes in files f1 and f2 and produce f3.

Description

Notes

9.32 mpps_evaluate

Name
mpps_evaluate –
Platforms
Unix, Windows
Synopsis
mpps_evaluate [–v] <mod | mgr> <test> <test file> <master>
Description
mpps_evaluate examines an MPPS file created by the MESA data server (*ds_dcm*). The MPPS object in the file is either a new MPPS object created by a modality or is a copy of an object forwarded by an MPPS manager. In this implementation, the evaluation is the same for both data sources.
The object is evaluated according to the the IHE MPPS cases (simple, unscheduled, group) as directed by the <*test*> argument. The application returns 0 if the MPPS object passes the evaluation and non-zero otherwise.
The <*master*> object provides a basis for evaluation. For example, this object will contain identifiers and code values taken from the Modality Worklist that are expected to be present in the test object. There are other attributes (such as image references), that we are unable to test.
Options

-v	Verbose mode
mod | mgr	Test output from modality (mod) or MPPS manager (mgr)
test	Test number (1 = simple, 2 = unscheduled, 3 = group)
test file	File with MPPS data to test
master	Master file considered gold standard

Notes
We should probably list all of the attributes tested in each case. That is probably the topic of another document.

9.33 mpps_merge

Name
mpps_merge – unsupported
Platforms
Unix, Windows
Synopsis
mpps_merge

Description

Notes

9.34 mpps_ncreate

Name
mpps_ncreate – deprecated
Platforms
Unix, Windows
Synopsis
mpps_ncreate

Description

Notes

9.35 mwl_query

Name
mwlquery –
Platforms
Unix, Windows
Synopsis
mwlquery [-a title] [-c title] [-f query] [-p] [-v] node port

-a title	Set the calling AE title
-c title	Set the called AE title
-f query	Name of a MESA DICOM file that contains a DICOM Modality Worklist query
node	Name of the host running the query server
port	TCP port number of server application

Description

Notes

9.36 naction

Name
naction – Send a DICOM N-Action Request and print result
Platforms
Unix, Windows
Synopsis
naction [-a title] [-c title] [-i ID] [-l] [-p] [-v] node port <SOP Class> file <SOP Instance>

-a title	Set the calling AE title
-c title	Set the called AE title
-i ID	Set the Action ID in the N-Action Request. Default value is 1.
-l	List the supported SOP Class synonyms. Perform no action.
-v	Verbose mode
node	Name of the host running the server application
port	TCP port number of server application
<SOP Class>	UID of a DICOM SOP class or a synonym recognized by this application.
file	Name of a file with the data set to be sent with N-Action Request
<SOP Instance>	A DICOM UID that identifies the SOP instance

Description

Notes

9.37 ncreate

Name
ncreate – Send a DICOM N-Create message to a server
Platforms
Unix, Windows
Synopsis
ncreate [-a title] [-c title] [-l] [-p] [-v] node port <SOP Class> file <SOP Instance>

-a title	Set the calling AE title
-c title	Set the called AE title
-l	List the supported SOP Class synonyms. Perform no action.
-v	Verbose mode
node	Name of the host running the server application
port	TCP port number of server application
<SOP Class>	UID of a DICOM SOP class or a synonym recognized by this application.
file	Name of a file with the data set to be sent with N-Create Request
<SOP Instance>	A DICOM UID that identifies the SOP instance

Description

Notes

9.38 nevent

Name
nevent – Send a DICOM N-Event Report Request message to a server
Platforms
Unix, Windows
Synopsis
nevent [-a title] [-c title] [-i ID] [-l] [-p] [-v] node port <SOP Class> file <SOP Instance>

-a title	Set the calling AE title
-c title	Set the called AE title
-i ID	Set the Event Type ID in the N-Event Report. Default value is 1.
-l	List the supported SOP Class synonyms. Perform no action.
-v	Verbose mode
node	Name of the host running the server application
port	TCP port number of server application
<SOP Class>	UID of a DICOM SOP class or a synonym recognized by this application.
file	Name of a file with the data set to be sent with N-Event Report
<SOP Instance>	A DICOM UID that identifies the SOP instance

Description

Notes

9.39 nset

Name
nset – Send a DICOM N-Set Request message to a server
Platforms
Unix, Windows
Synopsis
nset [-a title] [-c title] [-l] [-p] [-v] node port <SOP Class> file <SOP Instance>

-a title	Set the calling AE title
-c title	Set the called AE title.
-l	List the supported SOP Class synonyms. Perform no action.
-v	Verbose mode
node	Name of the host running the server application
port	TCP port number of server application
<SOP Class>	UID of a DICOM SOP class or a synonym recognized by this application.
file	Name of a file with the data set to be sent with N-Set Request
<SOP Instance>	A DICOM UID that identifies the SOP instance

Description

Notes

9.40 of_cancel

Name
of_cancel – unsupported
Platforms
Unix, Windows
Synopsis
of_cancel

Description

Notes

9.41 of_dcmps

Name
of_dcmps – Order Filler DICOM Personality Server, handles DICOM requests for Order Fillers
Platforms
Unix, Windows
Synopsis
of_dcmps [-a logdir][-b numbase] [-c title] [-d FAC] [-f] [-i] [-j] [-l level] [-m maxPDU] [-p] [-t trips] [-v] [-z] <port | file>

Description
of_dcmps is a data server for DICOM functions designed for Order Fillers. It supports the MWL and MPPS SOP classes as an SCP.
The description of this application is too complex for this document. This application is more fully described in the document *MESA Server Applications*.
Options

-a	Set the log directory; default is $MESA_TARGET/logs
-b	Set starting number for output files when run in file mode
-c	Set the AE title of this server
-d	Place one facility (DCM, DUL, SRV) in verbose mode
-f	Fork a new process for each connection
-i	Ignore some incorrect parameters in Association request
-j	Use thread model; spawn a new thread for each connection
-l	Set log level (0-4); default is 0: none
-m	Set the maximum received PDU in the Association RP to maxPDU
-p	Dump the Assocation RQ parameters
-t	Execute the main loop *trips* times (debugging tool)
-v	Place DUL and SRV facilities in verbose mode
-y	Print connection statistics
-z	Run in file mode. Process one MWL query from a file
port/file	Uses TCP/IP port or read MWL query from this file

Notes

9.42 of_hl7ps

Name
of_hl7ps – Order Filler HL7 personality server
Platforms
Unix, Windows
Synopsis
of_hl7ps [-a] [-b base] [-d definition] [-f] [-l level] [-s logDir] [-z dbName] <port | file>
Description
of_hl7ps is a data server for HL7 functions designed for Orders Fillers.
The description of this application is too complex for this document. This application is more fully described in the document *MESA Server Applications*.
Options

-a	Use analysis mode
-b	Set base directory for HL7 parsing rules; default is $MESA_TARGET/runtime
-d	Set suffix for HL7 parsing rules; default is ihe
-f	Use file mode; process one local file rather than read from network
-l	Set log level (0 = no logging, 4 = verbose)
-s	Set the directory for log files
-z	Set database name (default = ordfil)
port/file	TCP/IP port number or file to process

Notes

of_identifier

Name

of_identifier – create and print a new Order Filler identifier

Platforms

Unix, Windows

Synopsis

of_identifier <database> <identifier>

Description

of_identifier creates and prints a new Order Filler identifier. This is used in a scripting environment when a script needs to create such an identifier.

Options

database	Name of database to use	
identifier	Use one of:	
	accnum	Accession Number
	req_proc_id	Requested Procedure ID
	sps_id	Scheduled Procedure Step ID

Notes

9.43 of_mwl_cancel

Name
of_mwl_cancel – cancel MWL entries by removing removing them from MWL table
Platforms
Unix, Windows
Synopsis
of_mwl_cancel –p <plaordnum> <database>
Description
of_mwl_cancel removes all entries in the MWL table with the Placer Order Number specified by the user. This effectively cancels the order as seen from a Modality's point of view.
Options

-p	Set Placer Order Number; required switch
database	The name of the database to use

Notes

9.44 of_new

Name
of_new – unsupported
Platforms
Unix, Windows
Synopsis
of_new

Description

Notes

9.45 of_schedule

Name
of_schedule –
Platforms
Unix, Windows
Synopsis
of_schedule –m modality [-n <perf.phys.name>] [-p procedure] [-r] [-s <scheduled station name>] [-t AETitle] database
Description
of_schedule schedules one or more procedures on behalf of the Order Filler. If the –p switch is specified, only those orders with the appropriate procedure code are scheduled. Otherwise, all orders in the placer order table that are not labeled as "DONE" are scheduled.
In this context, scheduling is the act of taking an order and creating appropriate entries in the Modality Worklist table. One procedure may yield one or more Scheduled Procedure Steps. Each order in the Placer Order table is labeled "DONE" so they are not repeated.
Options

-m	Set modality value; required switch; no default
-n	Set Performing Physician Name; default is empty string
-p	If set, schedule this specific procedure only
-r	Repeat flag; reschedule even those procedures marked DONE
-s	Set Scheduled Station Name; default is station1
-t	Set AE Title for MWL entry; required switch; no default
database	Name of database to use

Notes

9.46 of_sched_procedure

Name
of_sched_procedure – unsupported
Platforms
Unix, Windows
Synopsis
of_sched_procedure

Description

Notes

9.47 open_assoc

Name
open_assoc – open a DICOM association and hold it open
Platforms
Unix, Windows
Synopsis
open_assoc [-a title] [-c title] [-p] [-s sleep] [-v] [-x UID] node port
Description
open_assoc opens a DICOM association with a server and holds the association open for 10 seconds or for the time specified with the –s switch. The default is to use the DICOM Verification class; that can be changed with the –x switch.
Options

-a	Application title of this application
-c	Called AP title to use during Association setup
-p	Print Association Response parameters
-s	Time to sleep before releasing association
-x	Request SOP class identified by UID rather than default verification
-v	Verbose mode for DUL/SRV facilities
node	Node name of server
port	Port number of server

Notes
This application is used to test that servers can handle multiple associations in parallel.

9.48 op_hl7ps

Name
op_hl7ps – Order Placer HL7 Personality Server, receive and process HL7 messages
Platforms
Unix, Windows
Synopsis
hl7_ps [-a] [-b base] [-d def] [-l level] [-s dir] [-v] [-w] [-z db] port
Description
op_hl7ps is a data server for HL7 functions designed for Orders Placers.
The description of this application is too complex for this document. This application is more fully described in the document *MESA Server Applications*.
Options

-a	Use analysis mode; capture and store HL7 messages in files
-b	Set base directory for HL7 parsing rules; default is $MESA_TARGET/runtime
-d	Set suffix for HL7 parsing rules; default is ihe
-l	Set log level (0 = no logging, 4 = verbose)
-s	Set directory for log files; default is $MESA_TARGET/logs
-v	Set verbose mode for HL7 message handler
-w	HL7 handler will capture HL7 input stream in a file
-z	Choose database name other than default (ordplc)
port	TCP/IP port number to use

Notes

9.49 op_order

Name
op_order – unsupported
Platforms
Unix, Windows
Synopsis
op_order

Description

Notes

9.50 op_send

Name
op_send – unsupported
Platforms
Unix, Windows
Synopsis
op_send

Description

Notes

9.51 orderplacer

Name
orderplacer – unsupported
Platforms
Unix, Windows
Synopsis
orderplacer

Description

Notes

9.52 sc_scp_association

Name
sc_scp_association – propose DICOM Associations on behalf of a Storage Commitment SCP
Platforms
Unix, Windows
Synopsis
sc_scp_association [-a title] [-c title] [-v] node port
Description
sc_scp_association proposes two DICOM Associations on behalf of a Storage Commitment SCP. This application is designed to test the reponse of a Storage Commitment SCU. This is in the context of the Storage Commitment Push Model.
In the first test, this application proposes an association and uses the default SCU role. Because the application under test is supposed to be the SCU, it should reject the presentation context for this SOP Class. (We also request the Verification SOP class, so the application under test should accept that SOP class but reject the presentation context for the Storage Commitment class).
In the second test, this application proposes an association and uses role negotiation to propose the SCP role for Storage Commitment. This application expects the application under test to accept that presentation context, but it must handle the SCP role negotiation properly.
If both tests are passed, this application returns 0. The application returns 1 if either test fails.
Options

-a	Application title of this application
-c	Called AP title to use during Association setup
-v	Verbose mode for DUL/SRV facilities
node	Node name of server
port	Port number of server

Notes
This application is used to test a Modality that implements the Storage Commitment SOP class. It may be unreasonable to expect a Modality to reject an "ill-formed" Storage Commitment presentation context. However, the Modality must be able to handle the SCP role negotiation found in test 2.
We have seen at least one device that failed test 2 because it received the SCP role and responded by accepting both the SCU and SCP roles. That contradicts the DICOM standard in that servers can only accept or reject what is presented; they should not accept other roles arbitrarily.

9.53 sc_scu_association

Name
sc_scu_association – propose DICOM Associations on behalf of a Storage Commitment SCU
Platforms
Unix, Windows
Synopsis
sc_scu_association [-a title] [-c title] [-v] node port
Description
sc_scu_association proposes two DICOM Associations on behalf of a Storage Commitment SCU. This application is designed to test the reponse of a Storage Commitment SCP. This is in the context of the Storage Commitment Push Model.
In the first test, this application proposes an association and uses the default SCU role. The system under test should accept this association and presentation context.
In the second test, this application proposes the Storage Commitment Push Model as an SCP. Because the system under test is expected to be an SCP, it should reject this presentation context. (We also request the Verification SOP class, so the application under test should accept that SOP class but reject the presentation context for the Storage Commitment class).
If both tests are passed, this application returns 0. The application returns 1 if either test fails.
Options

-a	Application title of this application
-c	Called AP title to use during Association setup
-v	Verbose mode for DUL/SRV facilities
node	Node name of server
port	Port number of server

Notes
This application is used to test an IHE Image Manager that implements the Storage Commitment SOP class. It may be unreasonable to expect an Image Manager to reject an “ill-formed” Storage Commitment presentation context. However, this test will demonstrate the role negotiation issues to the developer.

9.54 send_hl7

Name
send_hl7 – Send an HL7 message to a peer application.
Platforms
Unix, Windows
Synopsis
send_hl7 [-b base] [-d definition] [-n prefix] node port file [file...]

-b base Base directory which holds HL7 definition files. Default is /opt/mesa.
-d definition Extension to use when opening HL7 definition files. Default is “ihe”.
-n prefix Compute a new message control ID when sending message rather than using the value found in the file. Prefix will be placed before the newly computed value.
node Host name of the system running the server application
port TCP port number of the server application
file One or more files which contain HL7 messages to be sent

Description

Notes

9.55 sr_to_hl7

Name

sr_to_hl7 – convert a DICOM SR object to an HL7 ORU message

Platforms

Unix, Windows

Synopsis

sr_to_hl7 [-b base] [-d definition] [-t template] [-v] file1 file2

Description

sr_to_hl7 converts a DICOM SR object to an HL7 ORU message (ORU^R01). The process follows the recommendations found in the IHE Technical Framework. Multiple OBX segments are produced that describe some of the unique identifiers as well as contain the text of the report.

Options

-b	Set base directory for HL7 parsing rules; default is $MESA_TARGET/runtime
-d	Set suffix for HL7 parsing rules; default is ihe
-t	Set template file for MSH; required argument
-v	Verbose mode
file1	Input SR document
file2	Output HL7 message

Notes

The template file contains variables that should be set in the MSH segment of the output. The format of the file is

VARIABLE VALUE

The example below lists the variables and example values. Empty lines or lines that begin with # are ignored

```
# Comment
SENDING_APPLICATION      MESA_RPT_MGR
SENDING_FACILITY         EAST_RADIOLOGY
RECEIVING_APPLICATION    REPOSITORY
RECEIVING_FACILITY       XYZ
PROCESSING_ID            P
VERSION_ID               2.3.1
```

9.56 tcp_connect

Name
tcp_connect – open a TCP/IP connection to a server
Platforms
Unix, Windows
Synopsis
tcp_connect [-b base] [-d def] [-v] host port
Description
tcp_connect opens a TCP/IP connection with the designated server to determine if the server is running. The connection is closed immediately.
The application returns 0 if the connection is successful and 1 if not.
Options

-b	Set base directory for HL7 parsing rules; default is $MESA_TARGET/runtime
-d	Set suffix for HL7 parsing rules; default is ihe
-v	Verbose mode
host	Host name or IP address of server
port	TCP/IP port number of server

Notes

9.57 txt_to_hl7

Name
txt_to_hl7 – Convert a file from MESA text format to MESA HL7 format
Platforms
Unix, Windows
Synopsis
txt_to_hl7 [-b base] [-d def]
Description
txt_to_hl7 reads a description of an HL7 message from the standard input, creates an HL7 formatted stream and writes that stream to the standard output.
Options

-b	Set base directory for HL7 parsing rules; default is $MESA_TARGET/runtime
-d	Set suffix for HL7 parsing rules; default is ihe
txt file	Input text file
hl7 file	Output HL7 file

Notes

A: Appendix A: Text File Formats

Some CTN and MESA applications use a text file as input for creating objects or messages or for modifying those items. There are several different formats documented in this appendix.

CTN Simple Object Specification

Two CTN applications, *dcm_make_object* and *dcm_modify_elements*, use a simple text format for creating or modifying a DICOM object. Each line in the specification is of the form:

gggg eeee value

where *gggg* and *eeee* specify the group and element of the DICOM tag of the attribute. *value* is the value of the element. In this format, a value of "#" is special and indicates the attribute should be of zero length. All other strings are processed and are entered as the value. A line that begins with # is considered a comment.

This language only supports attributes at the top level of an object. That is, there is no support for sequences.

An example follows below:

```
# Comment line
0008 0050 A10205
0008 0080 Institution Name
0010 0010 Last^First
0010 0020 P5001
```

Note the spaces in the value for attribute 0008 0080 and the lack of any punctuation or delimiters in this format.

A.1: CTN Advanced Object Specification

Two different CTN applications, *dcm_create_object* and *dcm_modify_object*, use a more advanced language for creating or modifying a DICOM object. This language allows the applications to support sequences
The ASCII description of the DICOM object must conform to the syntax specified in the BNF notation below. A comment begins with "//" and continues to the end of the current text line.

```
obj_specification ->   element_list
element_list ->        element_list element | element
element ->             group_number element_number value
value ->               numeric_value | alphanumeric_value | quoted_string |
tag_value | multiple_values | sequence
tag_value ->           `<`group_number `,'  element_number `>'
multiple_value ->      `{` value_list `}' | `{` tag_list `}'
value_list ->          value_list `,' value | value
tag_list ->            tag_list tag_value | tag_value
sequence ->            sequence seq_item | seq_item
seq_item ->            `(` element_list `)'
```

Examples are given below.

```
// An example of single valued element
0018 1010 50                // patient's age in kilograms
// An example of multi-valued element
0054 0010 {1,1,1,1,1,2,2,2,2,2}              // NM isotope
vector
// An example of multi-valued element with quoted strings
0008 0008 {ORIGINAL, PRIMARY, "RECON GATED TOMO", EMISSION}
//image type
// An example of a tag type element with value multiplicity 1
0028 0009 <0054, 0010> // Frame increment pointer: single valued
// An example of a tag type element with value multiplicity 2
0028 0009 { // Frame increment pointer with multiplicity 2
      <0054, 0010>,   // index to isotope vector
     <0054, 0020>    // index to detector vector
}
// An example of a sequence type element with one item
0008 1110 ( // Referenced Study Sequence
  0008 1150 1.2.840.10008.5.1.4.1.1.20  // SOP class UID
  0008 1155 1.2.840.6839.1993.763.752369842.1 // Instance UID
)
// An example of a sequence type element with multiple items
0054 0062 ( // gated information sequence
   0018 1063 10    // frame time
   0018 1084 4     // intervals rejected
 )
(
   0018 1063 12    // frame time
  0018 1084 3     // intervals rejected
)
```

A.2: MESA Delta Object Specification

Several MESA applications transmit DICOM objects (composite or normalized) by reading the object from a file and applying a delta to the object before transmission. The delta is expressed in an ASCII file that is formatted as described below. Empty lines or lines that begin with '#' are ignored.
Each delta line begins with a two character code followed by other values as needed. The codes and other values are:

    AA gggg eeee <text string>
    AS gggg eeee
    AI ggg1 eee1 ggg2 eee2 <text string>

The AA command adds a single attribute at the top level of the object. The AS command adds an empty sequence at the top level of the object. The AI command adds one attribute within a sequence. For that AI operation, we can only construct a sequence with one set of sequence items. If you need to construct sequences with multiple items, you will need to use a different tool.