DHCP

HEALTH LEVEL SEVEN

(HL7)

USER MANUAL: TCP/IP SUPPLEMENT

Version 1.6*19

January 1999

Department of Veterans Affairs
V*IST***A** Software Development

Table of Contents

1. Patch HL*1.6*19 Release Notes

1.1. TCP/IP Transport Layer

As originally released, DHCP HL7 V. 1.6 supports transmission of HL7 messages between systems using the following transport mechanisms:

- Serial communications lines
- MailMan messages

Patch HL*1.6*19 adds an additional transport mechanism:

- Transmission Control Protocol/Internet Protocol (TCP/IP)

V*IST*A systems can now exchange HL7 messages over a TCP/IP network with both **V*IST*A** and non-**V*IST*A** systems.

Patch HL*1.6*19 supports *bi-directional* transmission of HL7 messages over TCP/IP, using the Minimal Lower Layer Protocol (MLLP) protocol, described later in the *TCP/IP LLP Implementation* chapter. Patch HL*1.6*19 implements bi-directional transmission as follows:

- The initiator of a transaction is the sending system.
- After the initial message is sent, the sending system waits for the expected response from the receiving system, which should be returned over the same open channel.
- Upon receipt of the response, the transaction is considered complete.
- If the sending system has another transaction in the queue to send, it is sent.
- At the completion of the last transaction, the connection is closed.
- The receiving system cannot initiate a new transaction of its own over the same open channel (it must open a new connection to do this.)

A number of **V*IST*A**'s HL7 transactions currently use VA MailMan as their transport layer; moving these transactions to TCP/IP logical links may be desirable to take advantage of TCP/IP transport layer features such as immediate commit acknowledgements.

1.2. View Transmission Log Option

Patch HL*1.6*19 introduces a new option, View Transmission Log. Using a VA FileMan Browser interface, you can query the transmission log to see pending or transmitted messages. You can filter the message view by date entered, message type and event type.

This new option replaces the former method of scanning globals directly to analyze message transmissions.

1.3. Performance Improvements

By restructuring the use of files, patch HL*1.6*19 reduces by at least 50% the disk hits incurred by the creation and transmission of HL7 messages.

Before patch HL*1.6*19, to create and transmit a message, the HL7 package incurred 5 major disk writes:

1. Store outgoing message text in File # 772 (HL7 Message Text).
2. Store a message ID and date/time in File # 773 (HL7 Message Administration).
3. When attaching a recipient, create a new stub entry in File # 772.
4. When attaching a recipient, create a new stub entry in File # 773.
5. When transmitting, create a new entry in File # 870 (HL Logical Link), combine text and recipient information from File # 772 and 773, attach a message header, and then transmit completely formed message.

Patch HL*1.6*19 consolidates the storage of non-TCP/IP message components in File #772 (HL7 Message Text) and expands the use of File # 773 (HL7 Message Administration) file for TCP/IP messages.

After patch HL*1.6*19, for all messages except those transmitted over TCP/IP, the HL7 package now incurs only 3 major disk writes:

1. Store outgoing message text, message ID and date/time in File # 772.
2. Store recipient information in File #772.
3. When transmitting, create a new entry in File # 870 (HL Logical Link), combines text and recipient information from File # 772, attach a message header, and then transmit completely formed message.

After patch HL*1.6*19, for all messages transmitted over TCP/IP, to create and transmit a message, the HL7 package now incurs only 2 major disk writes:

1. Store outgoing message text in File # 772.
2. Store recipient information in File #773.
3. When transmitting, transmit directly from information in File #772 and File #773.

A future patch to the HL7 package will result in all message transmissions (not just TCP/IP transmission) being performed directly out of file #772 and file #773, minimizing the number of disk hits and maximizing package performance.

1.4. V*IST*A HL7 Documentation Updates

The **V*IST*A** HL7 application is being updated through enhancement patches. Two supplemental manuals to the original documentation are provided (this manual for patch HL*1.6*19, and the *DHCP HL7 Developer Manual: Dynamic Addressing Supplement* for patch HL*1.6*14). Eventually, a single, consolidated, revised HL7 User Manual will be provided, combining all current and future revisions for both original HL7 User (site manager) and Developer manuals. Release will coincide with the release of future Messaging functionality, slated for FY99Q4.

1.5. Technical Information

1.5.1. New Routines

```
HLCSHDR1, HLCSRPT, HLCSRPT1, HLCSTCP, HLCSTCP1, HLCSTCP2, HLPAT19, HLTP3,
HLTP4 , HLTPCK2, HLTPCK2A
```

1.5.2. Data Dictionary Modifications

File # 869.1 HL LOWER LEVEL PROTOCOL TYPE

New Data:

- One record, TCP, has been added and will be merged into your current data.

File # 869.2 HL LOWER LEVEL PROTOCOL PARAMETER

New Fields:

- TCP/IP ADDRESS (F) [400;1]
- TCP/IP PORT (NJ5,0) [400;2]
- TCP/IP SERVICE TYPE (S) [400;3]
- PERSISTENT (S) [400;4]
- STARTUP NODE (P14.7) [400;6]

File # 771.6 HL7 MESSAGE STATUS

New Data:

- Several new statuses have been added and will overwrite your current data.

File # 771.7 HL7 ERROR MESSAGE

New Data:

- Several new errors messages have been added and will overwrite your current data.

File # 772 HL7 MESSAGE TEXT

Changed Field Definition:

- The .01 field, DATE/TIME ENTERED, has been changed from a pointer to a date/time field.

File #773 HL7 MESSAGE ADMINISTRATION

Changed Field Definition:

- The .01 field, DATE/TIME ENTERED, has been changed from a date/time to a pointer to file 772.

New Fields:

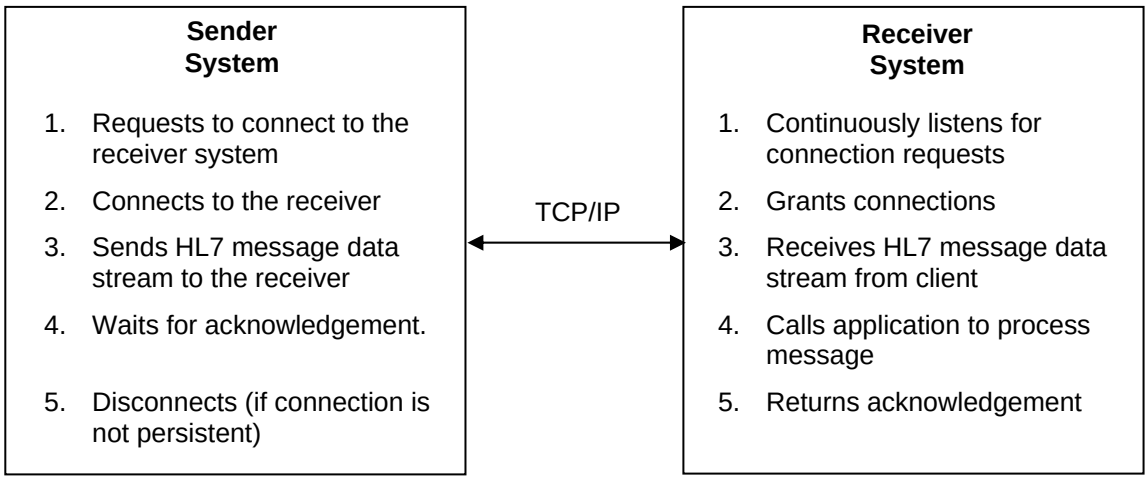
- TRANSMISSION TYPE (RSI), [0;3]
- PRIORITY (S), [0;4]
- HEADER TYPE (S), [0;5]
- ACKNOWLEDGEMENT TO (P773'), [0;6]
- LOGICAL LINK (P870'), [0;7]
- SUBSCRIBER PROTOCOL (P101'), [0;8]
- SECURITY (F), [0;9]
- DON'T PURGE (S), [2;1]
- CONTINUATION POINTER (F), [1;1]
- SENDING APPLICATION (P771'), [0;11]
- RECEIVING APPLICATION (P771'), [0;12]
- MESSAGE TYPE (P771.2'), [0;13]
- EVENT TYPE (P779.001'), [0;14]
- STATUS (RP771.6'), [P;1]
- STATUS UPDATE DATE/TIME (D), [P;2]
- ERROR MESSAGE (F), [P;3]
- ERROR TYPE (P771.7'), [P;4]
- DATE/TIME PROCESSED (D), [S;1]
- MSH (Multiple-773.01), [MSH;0]
- .01 MSH (WL), [0;1]

2. How to Set Up TCP/IP Interfaces

2.1. Client and Server Roles in HL7 over TCP/IP

Two separate sets of M code define the roles of client and server over TCP/IP channels:

- TCP Client = Sender = HL7 Server Protocol (Event Driver)
- TCP Server = Receiver = HL7 Client Protocol (Subscriber)



2.2. Requirements for Sending System

If the sending system is a **V*IST*A** system, the sending application (as defined in the Interface Workbench on the sending **V*IST*A** system) must have:

- A server (event driver) protocol set up for each message transaction type.
- A client (subscriber) protocol set up for each destination for each message transaction type.
- HL LOGICAL LINK entries set up for each destination system, with all appropriate information including IP address and port.

All messages generated through the server (event driver) protocol will be directed to all appropriate client (subscriber) protocols, including receivers whose HL LOGICAL LINK identifies them as connected through TCP/IP.

2.3. Requirements for Receiving System

If the receiving system is not a **V*IST*A** system, it must support bi-directional TCP/IP transmission, with a TCP/IP listener process that responds to connection requests.

If the receiving system is a **V*IST*A** system, the receiving application (as defined in the Interface Workbench on the receiving **V*IST*A** system) must have:

- A server (event driver) protocol set up for each transaction type.
- A client (subscriber) protocol set up for each transaction type.
- A running TCP/IP listener. The TCP/IP listener is a process that listens for TCP/IP connection requests on a particular IP address and port. The listener process must be set up by the receiving site to be actively listening.

2.4. TCP/IP Listener Types

Two types of TCP/IP listeners are supported by the HL7 package:

- Single-Threaded
- Multi-Threaded

2.4.1. Single-Threaded Listener

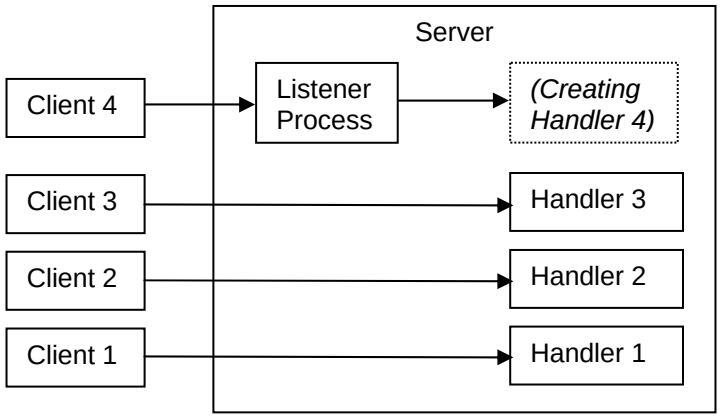
Single-threaded listeners are best suited for connecting to a specific device over TCP/IP, such as a single Commercial Off-The-Shelf (COTS) system. The HL7 package's single-threaded listener is implemented as an M process running on the server. It waits for connection requests and handles all processing for each connection directly without spawning another job.

The drawback to single-threaded listeners is that if the listener is busy processing an incoming HL7 message during an active connection, other connection requests fail.

For instructions on how to set up single-threaded TCP/IP listeners, please see the "TCP/IP Listener Setup" chapter.

2.4.2. Multi-Threaded Listener

Multi-threaded listeners are useful when multiple connection requests come to a single port from many devices or systems. An example of this scenario is the use of HL7 messaging by the Clinical Information Resource Network (CIRN). Because multi-threaded listeners spawn off separate handlers for each client connection request, they enable multiple concurrent connections.



For instructions on how to set up multi-threaded TCP/IP listeners, please see the "TCP/IP Listener Setup" chapter.

2.5. How to Set Up TCP/IP Logical Links

Each client (subscriber) protocol must point to a particular HL LOGICAL LINK entry, identifying the destination system to send the message to. Use the HL7 package's Interface Workbench to set up logical links.

For client (subscriber) protocols that need to connect to destination systems over TCP/IP, you need to set up a logical link for each destination system, with LLP Type set to TCP. Other fields in the logical link describe other characteristics of the destination system, including its TCP/IP address and listener port.

You must also define logical links for each TCP/IP listener on the current system. At a site, this is typically the responsibility of IRM. For more information see the "TCP/IP Listener Setup" chapter.

Note: HL LOGICAL LINK entries are not tied to any one protocol, and so can be reused by other protocols.

2.5.1. How to Set Up TCP/IP Logical Links

Use the Interface Workbench to create Logical Link entries for systems other than the current system. This link stores the path to a target system. The following field settings are appropriate for TCP/IP destination logical links:

LLP Type:	TCP
TCP/IP address:	Address of target system
TCP/IP Port:	Port the target system is listening on
TCP/IP Service Type:	CLIENT
Persistent:	Null if link is transient, YES if link should remain open always

```
HL-7 Interface Workbench      Sep 03, 1998 13:16:55          Page:    15 of   27
                                Currently Defined
                                  Logical Links
================================================================================

(21) VAWPX
 LLP Parameter: VAWPX TCP                     TCP/IP Address: 152.131.120.201
      LLP Type: TCP  (T)                        TCP/IP Port: 5000
    Queue Size: <DEFAULT>                TCP/IP Service Type: CLIENT
   Institution: <NONE>                            Persistent: <DEFAULT>
        Domain: <NONE>                          Startup Node: <NONE>
     Autostart: <DEFAULT>
```

2.5.1.1. Starting TCP/IP Logical Links

Starting up a Logical Link (with the START LLP option) is all you need to do to enable exchange of messages over the link. This starts up a process specific to the logical link that runs until you shut down the logical link.

No outbound filers are used for TCP/IP logical links. Instead, the role of the logical link process is expanded. Besides transmitting messages, the logical link process itself also detects whether there are messages to send for the logical link.

A consequence of this shift is that, for the time being, if you have 150 TCP/IP logical links for 150 destinations, 150 logical link processes will run concurrently. A future patch will address this issue, ideally before any system has near that many TCP/IP logical links. The future patch will introduce a queue manager that will start TCP/IP logical link processes only when they are needed to transmit outgoing messages.

2.5.2. New Logical Link Fields

Patch HL*1.6*19 introduces new fields to the HL LOWER LEVEL PROTOCOL PARAMETER file (#869.2) for TCP/IP logical links:

Field#	Name	Description
400.01	TCP/IP ADDRESS	TCP/IP address of the target system. Leave blank if describing a listener on the current system.
400.02	TCP/IP PORT	TCP/IP port to connect to (target system) or listen on (current system).
400.03	TCP/IP SERVICE TYPE	• 'CLIENT (SENDER)': Indicates that this Logical Link connects to a target system, with the current system acting as the sender. • 'SINGLE LISTENER': Designates that the current system is a server (listener), using a single threaded listener. • 'MULTI LISTENER': Designates that the current system is a server (listener), using a multiple threaded listener.
400.04	PERSISTENT	For TCP/IP SERVICE TYPE of Client only. Enter 'YES' if this connection needs to remain open even if there are no messages to send. A setting of 'YES' is appropriate for sending high volumes of messages to a COTS device. The connection remains open until either side disconnects it. On the **V*IST*A** side, disconnection is by shutting-down the logical link.
400.06	STARTUP NODE	Applies only if the current system is a listener, and is running dual TaskMan in a DCL context (OpenVMS systems only). If one or more clients can only connect to a single IP address (i.e., a particular node) you can use this field to force links (started by TaskMan) to start up on that particular node.

2.6. Acknowledgements Over TCP/IP

In HL7 versions 2.2 and above, Enhanced Mode differentiates between *commit accept acks* and *application acks*. The acknowledgement scheme is defined in the message header. The following protocol settings define the acknowledgement scheme in the message headers for **V*ISTA*** HL7:

- ACCEPT ACKNOWLEDGEMENT
- APPLICATION ACKNOWLEDGEMENT

Acknowledgements are performed as follows, based on these two settings:

Accept Ack [MSH-15]	Application Ack [MSH-16]	Ack Mode	Behavior
null	null	Original	Application ack
NE	AL	Original	Application ack
AL	AL	Enhanced	Commit ack, application ack (i.e., enhanced mode full two-phase commit)
AL	NE	Enhanced	Commit ack, no application ack
NE	NE	Enhanced	No commit ack, no application ack

2.6.1. Original Mode Acknowledgements

In Original mode, the receiving application is always expected to return a response (i.e., an application acknowledgement). This response may or may not contain data, but always contains an ack. For example, a query response will contain both data and an ack message.

Expected Behavior, Original Mode Acknowledgements over TCP/IP

Sending System	Receiving System
1. Open Connection. 2. Send Message.	
	3. Process Message. 4. Send Query Response or General Ack (AA, AE, AR).
5. Process Response. If not a positive ack, retransmit (go to step 2.) 6. Drop Connection.	

2.6.2. Enhanced Mode, Full Two-Phase Commit

For Enhanced mode acknowledgements with full two-phase commit, over TCP/IP, the COMMIT ACK should always be immediate and over the current connection. The application acknowledgement should always be deferred, and performed as a separate transaction. This means that the application ack is always deferred and the final "commit" is optional.

Expected Behavior, Enhanced Mode Two-Phase Commit over TCP/IP

Sending System	Receiving System
Phase I	
1. Open Connection. 2. Transmit Message.	
	3. Receive Message. 4. Validate Header. 5. Store on local system. 6. Transmit "Commit Accept Ack" (MSA contains 'CA').
7. Receive Commit Ack. 8. Process/validate Commit Ack. If not a positive commit ack, retransmit (go to step 2.) 9. Drop Connection.	
Phase II	
	1. Open Connection. 2. Transmit Application Response. Acknowledgment mode in header must be AL/NE or NE/NE. 3. If Ack mode of application response is NE/NE, drop connection.
4. Receive Application Response. 5. Validate Header. 6. Store on local system. 7. If MSH-15='AL', transmit "Commit Accept Ack" (MSA contains 'CA').	
	8. Receive Commit Accept Ack (if any). 8. Process/Validate Commit Accept Ack. If not a positive commit ack, retransmit (go to step 2.) 10. Drop Connection.

2.6.3. Addressing Deferred Acknowledgements

When you process a message, the way to send an application acknowledgement is calling GENACK^HLMA1. Among other things, GENACK^HLMA1 handles the details of where to address the response *to*, as follows:

1. During application processing of a message, the event driver and subscriber protocols involved in the transaction are available in HL("EID") and HL("EIDS").
2. The application calls GENACK^HLMA1 to send an application acknowledgement. It should pass HL("EIDS") as the parameter for the transaction's subscriber protocol.
3. GENACK^HLMA1 uses the subscriber protocol's logical link as the destination to return the response to.

This method works fine for traditional, point-to-point interfaces where the relationship between sender and receiver is defined in advance. However, with the introduction of *dynamic addressing*, transactions occur where the sender and receiver are *not* defined in advance. This poses a problem in certain situations.

For dynamically addressed messages over TCP/IP where original mode is used, returning application acknowledgements is *not* a problem, because the response is returned over the open TCP/IP channel.

However, for *enhanced mode, two-phased commit transactions* over TCP/IP, where the response is deferred, the method used by the GENACK^HLMA1 to address responses doesn't work. While GENACK^HLMA1 still sends the application acknowledgement back over the single logical link attached to the transaction's subscriber protocol, that logical link may have no correspondence with the actual originating system.

A future patch will support HL7 V. 2.3-style message headers, where all the information needed to address a response to a message is contained in that message's header (rather than in the advance interface setup). This more *flexible* arrangement will allow GENACK^HLMA1 to properly address deferred application acknowledgements for interfaces that are not point-to-point. Until this issue is resolved, however, there are two workarounds:

- Use original mode, which skips the commit acknowledgement and instead returns the application acknowledgement over the open TCP/IP channel.
- If the sender is a **V*IST*A** facility, and if that facility's institution number is in the original message's facility field, use the dynamic addressing features released in patch HL*1.6*14 to resolve the proper logical link and dynamically address the response.

2.7. Exporting TCP/IP Logical Links

The Kernel Installation and Distribution System (KIDS) allows you to export HL LOGICAL LINK and HL LOWER LEVEL PROTOCOL PARAMETER entries. You will, however, need to instruct sites to modify certain site-specific fields in any HL LOWER LEVEL PROTOCOL PARAMETER entries you export.

1. Export all HL LOWER LEVEL PROTOCOL PARAMETER entries with TCP/IP SERVICE TYPE set to CLIENT (SENDER), and TCP/IP ADDRESS defined. This exports all logical links as client connections to target systems.
2. Instruct sites to change their own logical link from client to server. To do this, they should change the HL LOWER LEVEL PROTOCOL PARAMETER settings for their own site as follows:
   - Delete the TCP/IP ADDRESS for their own site.
   - Change the TCP/IP SERVICE TYPE setting for their own site to SINGLE LISTENER or MULTI LISTENER (depending on what type of listener the site will use).

3. TCP/IP Listener Setup

3.1. How to Set Up Single-Threaded Listeners

Single-threaded listener mode for TCP/IP messaging is available for all currently supported M operating systems:

- Caché on NT
- DSM for OpenVMS

To set up single-threaded listeners, simply define an entry in the HL LOGICAL LINK file for each single-threaded listener. No additional setup is required.

3.1.1. Logical Link Setup

Using the Interface Workbench, create a Logical Link entry for the single threaded listener. The following field settings are appropriate for single-threaded listeners:

LLP Type:	TCP
TCP/IP address:	DSM for OpenVMS: null; Caché for NT: IP address of listener system
TCP/IP Port:	Port to listen on.
TCP/IP Service Type:	SINGLE LISTENER
Persistent:	Null
Startup Node:	(set only for OpenVMS systems running dual TaskMan.)

```
HL-7 Interface Workbench      Sep 03, 1998 13:16:55          Page:    15 of   27
                               Currently Defined
                                 Logical Links
================================================================================

(22) A6A LISTENER
 LLP Parameter: A6A5000                      TCP/IP Address: <NONE>
      LLP Type: TCP  (T)                        TCP/IP Port: 5000
    Queue Size: <DEFAULT>                TCP/IP Service Type: SINGLE LISTENER
   Institution: <NONE>                           Persistent: <DEFAULT>
        Domain: <NONE>                         Startup Node: <NONE>
     Autostart: <DEFAULT>
```

3.1.2. How to Start and Stop the Listener

To start single-threaded listeners, use the Start LLP option. Choose the Logical Link entry you defined for the listener. Typically you would run the link in the background. To stop the listener, use the Stop LLP option.

3.2. Multi-Threaded Listener Setup: Caché on NT

Kernel patch XU*8*78 provides a multi-threaded listener for TCP/IP messaging for Caché on NT systems.

3.2.1. Logical Link Setup

Using the Interface Workbench, create a Logical Link entry for the multi-threaded listener. The following field settings are appropriate for multi-threaded listeners for Caché on NT:

LLP Type:	TCP
TCP/IP address:	IP address of listener system
TCP/IP Port:	(the port to listen on)
TCP/IP Service Type:	MULTI LISTENER
Persistent:	null
Startup Node:	(set only for OpenVMS systems running dual TaskMan.)

```
HL-7 Interface Workbench      Sep 03, 1998 13:16:55          Page:    15 of   27
                               Currently Defined
                                 Logical Links
================================================================================

(23) A6A UCX LISTENER
 LLP Parameter: A6A5000                      TCP/IP Address: <NONE>
      LLP Type: TCP  (T)                        TCP/IP Port: 5000
    Queue Size: <DEFAULT>                TCP/IP Service Type: MULTI LISTENER
   Institution: <NONE>                           Persistent: <DEFAULT>
        Domain: <NONE>                         Startup Node: <NONE>
     Autostart: <DEFAULT>
```

3.2.2. How to Start and Stop the Listener

To start the multi-threaded listener for Caché on NT systems, use the Start LLP option. Choose the Logical Link entry you defined for this listener. Typically you would run the link in the background. To stop the listener, use the Stop LLP option.

3.3. Multi-Threaded Listener Setup: DSM for OpenVMS

For DSM for OpenVMS, multi-threaded listeners are implemented externally, using DSM for OpenVMS's UCX (also known as Digital TCP/IP Services for OpenVMS). UCX permits multiple TCP/IP clients to connect and run as concurrent processes, up to the limits established by the system. UCX listens on a particular port, and launches a specified HL7 handler process for each client connection.

For the UCX HL7 handler process, you need to create:

- an OpenVMS account
- a home directory
- a DCL (Digital Command Language) login command procedure

For the sake of the example in this chapter, the HL7 handler account name is HLSEVEN, the home directory is [HLSEVEN] and the DCL login command procedure is named HLSEVEN.COM.

3.3.1. Logical Link Setup

Using the Interface Workbench, create a Logical Link entry for the multi-threaded listener. The following field settings are appropriate for multi-threaded listeners implemented through UCX:

LLP Type:	TCP
TCP/IP address:	null
TCP/IP Port:	(the port to listen on)
TCP/IP Service Type:	MULTI LISTENER
Persistent:	null
Startup Node:	(set only for OpenVMS systems running dual TaskMan.)

Note: If you are *converting an existing logical link* to become a logical link for the UCX TCP/IP listener, rather than creating a new one, follow this procedure:

1. First, stop the logical link.
2. Next, use the "Clear a Queue of all Entries" option on the logical link. This resets all message counters and flags, ensuring the queue is initialized correctly.
3. Now use the Interface Workbench to change the logical link to TCP/IP.

```
HL-7 Interface Workbench      Sep 03, 1998 13:16:55          Page:    15 of   27
                                Currently Defined
                                  Logical Links
================================================================================

(23) A6A UCX LISTENER
 LLP Parameter: A6A5000                      TCP/IP Address: <NONE>
      LLP Type: TCP  (T)                       TCP/IP Port: 5000
    Queue Size: <DEFAULT>                TCP/IP Service Type: MULTI LISTENER
   Institution: <NONE>                          Persistent: <DEFAULT>
        Domain: <NONE>                        Startup Node: <NONE>
     Autostart: <DEFAULT>
```

3.3.2. OpenVMS Account Setup for HL7 Handler

The easiest way to configure an OpenVMS account to be an HL7 handler is to copy most of the parameters from VA MailMan TCP account. To do this:

1. Determine an unused User Identification Code (UIC), typically in the same group as other DSM for OpenVMS accounts.
2. Using the OpenVMS Authorize utility, copy the XMINET account to a new HLSEVEN account with the unused UIC. You must have SYSPRV to do this.
3. Make sure that the account settings for the new HLSEVEN account are the same as in the example below, or, if they are different, that the impact of the different settings is acceptable for your system. In particular, make sure that the DisCtlY, Restricted and Captive flags are set for security reasons.

3.3.2.1. Account Setup Example (Contains Recommended Settings)

```
$SET DEF SYS$SYSTEM
$MC AUTHORIZE
UAF> SHOW XMINET

Username: XMINET                          Owner:  DSM
Account:                                      UIC:    [50,44] ([XMINET])
CLI:      DCL                                 Tables: DCLTABLES
Default:  SYS$SYSDEVICE:[XMINET]
LGICMD:   NL:
Flags:  DisCtlY Restricted Captive
Primary days:   Mon Tue Wed Thu Fri
Secondary days:                       Sat Sun
Primary   000000000011111111112222  Secondary 000000000011111111112222
Day Hours 012345678901234567890123  Day Hours 012345678901234567890123
Network:  ##### Full access ######            ##### Full access ######
Batch:    -----  No access  ------            -----  No access  ------
Local:    -----  No access  ------            -----  No access  ------
Dialup:   -----  No access  ------            -----  No access  ------
Remote:   -----  No access  ------            -----  No access  ------
Expiration:   (none)    Pwdminimum:  6   Login Fails:     0
Pwdlifetime:  90 00:00    Pwdchange:      (pre-expired)
```

```
Last Login:   (none) (interactive), 10-FEB-1998 15:30 (non-interactive)
Maxjobs:         0  Fillm:       500  Bytlm:       100000
Maxacctjobs:     0  Shrfillm:      0  Pbytlm:           0
Maxdetach:       0  BIOlm:       150  JTquota:       4096
Prclm:           8  DIOlm:        18  WSdef:         1344
Prio:            4  ASTlm:       176  WSquo:         2688
Queprio:         4  TQElm:        10  WSextent:     65536
CPU:        (none)  Enqlm:      3000  Pgflquo:     100000
Authorized Privileges:
  NETMBX    OPER      SHARE     TMPMBX
Default Privileges:
  NETMBX    OPER      SHARE     TMPMBX

UAF> COPY XMINET HLSEVEN/UIC=[51,45]
%UAF-I-COPMSG, user record copied
%UAF-W-DEFPWD, copied or renamed records must receive new password
%UAF-I-RDBADDMSGU, identifier HLSEVEN value [000051,000045] added to
rights database

UAF> SHOW HLSEVEN

Username: HLSEVEN                          Owner:  DSM
Account:                                   UIC:    [51,45] ([HLSEVEN])
CLI:      DCL                              Tables: DCLTABLES
Default:  SYS$SYSDEVICE:[XMINET]
LGICMD:   NL:
Flags:  DisCtlY Restricted Captive
Primary days:   Mon Tue Wed Thu Fri
Secondary days:                      Sat Sun
Primary   000000000011111111112222  Secondary 000000000011111111112222
Day Hours 012345678901234567890123  Day Hours 012345678901234567890123
Network:  ##### Full access ######            ##### Full access ######
Batch:    -----  No access  ------            -----  No access  ------
Local:    -----  No access  ------            -----  No access  ------
Dialup:   -----  No access  ------            -----  No access  ------
Remote:   -----  No access  ------            -----  No access  ------
Expiration:    (none)    Pwdminimum:  6   Login Fails:     0
Pwdlifetime:   90 00:00    Pwdchange:      (pre-expired)
Last Login:    (none) (interactive),           (none) (non-interactive)
Maxjobs:         0  Fillm:       500  Bytlm:       100000
Maxacctjobs:     0  Shrfillm:      0  Pbytlm:           0
Maxdetach:       0  BIOlm:       150  JTquota:       4096
Prclm:           8  DIOlm:        18  WSdef:         1344
Prio:            4  ASTlm:       176  WSquo:         2688
Queprio:         4  TQElm:        10  WSextent:     65536
CPU:        (none)  Enqlm:      3000  Pgflquo:     100000
Authorized Privileges:
  NETMBX    OPER      SHARE     TMPMBX
Default Privileges:
  NETMBX    OPER      SHARE     TMPMBX

UAF> MOD HLSEVEN/DIR=[HLSEVEN]
%UAF-I-MDFYMSG, user record(s) updated
UAF> EXIT

%UAF-I-DONEMSG, system authorization file modified
%UAF-I-RDBDONEMSG, rights database modified
```

3.3.3. Home Directory Setup for HL7 Handler Account

You need to create a home directory for the HL7 handler account. This directory will house the DCL command procedure that is executed whenever a client connects, as well as log files. Make sure that the owner of the directory is the HLSEVEN account.

For example, to create a home directory named [HLSEVEN] with ownership of HLSEVEN:

```
$CREATE/DIR [HLSEVEN]/OWNER=HLSEVEN
```

3.3.4. DCL Login Command Procedure for HL7 Handler

Create a DCL command procedure like the following, in the home directory for the handler account. Make sure the command procedure file is owned by the HL7 handler account.

1. Adjust the DSM command line (environment, UCI and volume set) so that it is correct for your system.
2. If access control is enabled, ensure that the HLSEVEN account has access to this UCI, volume set and routine.
3. Replace 999 with the internal entry number of the logical link you created (in the HL LOGICAL LINK file) to be invoked by the listener-spawned process.

```
$!HLSEVEN.COM - for incoming connect requests
$!--------------------------------------------------------------------
$ set noon                        !Don't stop
$ set verify
$ purge/keep=2 sys$login:*.*
$ set proc/priv=(share)           !Required to use the MBX device
$ x=f$trnlnm("sys$net")           !This is our MBX device
$ write sys$output x              !This can be viewed in the log file
$ set nover                      !Turn off verify
$!--------------------------------------------------------------------
$! **Be sure this command line is correct for your system
$! **and if access control is enabled that this account has
$! **access to this uci,vol & routine.  The number 999 should be replaced
$! **with the internal entry number in file 870 for this Logical Link
$!                                                    /
$ dsm/environ=MGRISC/uci=VAH/vol=ROU/data="''x'^999" EN^HLCSTCP
$!--------------------------------------------------------------------
$ logout/brief
```

3.3.5. UCX Service

Once you've created the HL7 handler, you can create the UCX service that will listen for connections and launch the HL7 handler. Choose the port it should listen on, and the user and command file to invoke when a connection is received.

For a test account, you can set up a UCX service for testing, and it can use the same OpenVMS account and directory as the production UCX service. Just create a different DCL command file with the UCI and volume set of the test account.

(Note: Since UCX is node specific, make sure you are on the correct node.)

```
$UCX
UCX> SET SERVICE HLSEVEN/USER=HLSEVEN/PROC=HLSEVEN /PORT=5000-
_UCX> /PROTOCOL=TCP/REJECT=MESSAGE="All channels busy" -
_UCX> /LIMIT=50/FILE=SYS$SYSDEVICE:[HLSEVEN]HLSEVEN.COM

UCX> SHO SERVICE HLSEVEN/FULL

Service: HLSEVEN
                          State:     Disabled
Port:             5000    Protocol:  TCP             Address:  0.0.0.0
                          User_name: not defined     Process:  HLSEVEN

UCX> ENABLE SERVICE HLSEVEN
UCX> SET CONFIG ENABLE SERVICE HLSEVEN
UCX> SHO SERVICE/FULL HLSEVEN

Service: HLSEVEN
                          State:     Enabled
Port:             5000    Protocol:  TCP             Address:  0.0.0.0
Inactivity:          5    User_name: HLSEVEN         Process:  HLSEVEN
Limit:              50    Active:      0             Peak:       0

File:         SYS$SYSDEVICE:[HLSEVEN]HLSEVEN.COM
Flags:        Listen

Socket Opts:  Rcheck Scheck
 Receive:            0    Send:               0

Log Opts:     None
 File:        not defined

Security
 Reject msg:  All channels busy

 Accept host: 0.0.0.0
 Accept netw: 0.0.0.0

UCX> SHO CONFIG ENABLE SERVICE

Enable service
     FTP, FTP_CLIENT, HLSEVEN, MPI, TELNET, XMINETMM
UCX> EXIT
```

3.3.6. Access Control List (ACL) Issues

Some sites use DSM's ACL feature, which gives explicit access to each OpenVMS account that needs to enter that DSM environment. If your site is using ACL, you should set up the HLSEVEN account with APPLICATION access, and then specify the Volume, UCI and routine that the HLSEVEN user has authorization to access.

An example of setting this level of access for an HLSEVEN account is provided below:

```
$ DSM /MAN ^ACL

Environment Access Utilities

    1.  ADD/MODIFY USER                   (ADD^ACL)
    2.  DELETE USER                       (DELETE^ACL)
    3.  MODIFY ACTIVE AUTHORIZATIONS      (^ACLSET)
    4.  PRINT AUTHORIZED USERS            (PRINT^ACL)

Select Option > 1  ADD/MODIFY USER

OpenVMS User Name:   >   HLSEVEN

ACCESS MODE    VOL       UCI       ROUTINE
-----------    ---       ---       -------

No access rights for this user.

Access Mode ([M]ANAGER, [P]ROGRAMMER, or [A]PPLICATION):   >   A
Volume set name:   >   VAH
UCI:   >   ROU
Routine:   >   HLCSTCP
Routine:   >   <RET>
UCI:   >   <RET>
Volume set name:   >   <RET>
Access Mode ([M]ANAGER, [P]ROGRAMMER, or [A]PPLICATION):   >   <RET>

USER           ACCESS MODE    VOL       UCI       ROUTINE
----           -----------    ---       ---       -------

HLSEVEN        APPLICATION    VAH       ROU       HLCSTCP

OK to file?   <Y>   <RET>

OpenVMS User Name:   >   <RET>

OK to activate changes now?   <Y>   <RET>

Creating access authorization file:  SYS$SYSDEVICE:[DSMMGR]DSM$ACCESS.DAT.
```

3.3.7. Controlling the Number of Log Files Created by UCX

The new HLSEVEN service will automatically create log files (UCX does this and it cannot be prevented) in the HLSEVEN directory named HLSEVEN.LOG;xxx where 'xxx' is a file version number. New versions of this file will be created until that file version number reaches the maximum number of 32767. In order to minimize the number of log files created, you may want to initially rename this log file to the highest version number with the command:

```
$ RENAME disk$:[HLSEVEN]HLSEVEN.LOG; disk$:[HLSEVEN]HLSEVEN.LOG;32767
```

Alternatively, you can set a limit on the number of versions of the log file that can concurrently exist in the HLSEVEN directory:

```
$ SET FILE /VERSION_LIMIT=10 disk$:[HLSEVEN]HLSEVEN.LOG;
```

Note that you would not want to limit the number of versions of the log file until you know that your HLSEVEN service is working correctly; you can use the log file to help diagnose problems with the service/account.

3.3.8. How to Start and Stop the Listener

Although the multi-threaded listener for OpenVMS systems requires the setup of a logical link, you never actually start or stop this logical link with HL7 package options. Instead, because the listener is implemented as a UCX service, start it up and shut it down by using the generic tools UCX provides to start up and shut down any UCX service.

4. Tracking HL7 Message Transmissions

The new View Transmission Log option makes it easier to look at which HL7 messages have been transmitted (incoming and outgoing) or are pending for transmission. This new option replaces the more tedious former process of looking through globals.

4.1. For TCP/IP Messages Only

The View Transmission Log option tracks messages transmitted over TCP/IP logical links only, at this time. The option takes advantage of new, simplified file structures in the HL7 package, currently used only for TCP/IP message transmission. Eventually, messages transmitted over non-TCP/IP logical links will be converted to use the new file structure, and will then be viewable in the View Transmission Log option.

4.2. Choosing What Messages to View

Choose the 'View Transmission Log' option from the HL7 Communications Server Menu.

```
                              Search Transmission Log



     Select one of the following:

          A         All Completed Transmissions
          E         Event Type Search (Completed)
          L         Logical Link Search (Completed)
          P         Pending Transmissions
          Q         Quit (also uparrow, or <RETURN>)

Selection:
```

Choose whether to look at all transmissions, transmissions for a particular event type, transmissions over a particular logical link, or all pending transmissions.

For each choice, you are able to further filter the list of messages that are displayed:

Option	**Filter By**
All Completed Transactions	Date Range
Event Type Search (Completed)	Date Range, Event Type, Message Type
Logical Link Search (Completed)	Date Range, Logical Link
Pending Transmissions	(all)

4.3. Viewing Message Summaries

The View Transmission Log option displays summaries of all matching messages in VA FileMan's Browser. Messages are sorted by DATE/TIME PROCESSED value in HL7 Message Administration file (#773). For outgoing messages, this is the date/time that the message was transmitted; for incoming messages, this is the date/time that the receiving application processed the message.

```
MESSAGE ID #         Dte/TimeEntr'd Log Link   Msg:Evn IO Sndg Apl Rcvr Apl HDR
1                    071698.075746  INDY-TCP   ADT:A01 IN MPI-STAR RG CIRN  MSH^
3                    071698.075746  INDY-TCP   ACK:A01 OT RG CIRN  MPI-STAR MSH^
4                    072998.132139  INDY-TCP   ADT:A01 IN MPI-STAR RG CIRN  MSH^
6                    072998.132139  INDY-TCP   ACK:A01 OT RG CIRN  MPI-STAR MSH^




Col>   1  |<PF1>H=Help <PF1>E=Exit| Line>      4 of 4      Screen>   1 of 1
```

The following information is displayed for each message:

MESSAGE ID#	Unique Message ID number assigned by the HL7 package to the message.
Dte/TimeEntr'd	Exact time and date the message was first handed to HL7.
Log Link	Logical Link on which the message is to be, or was transmitted.
Msg:Evn	Message Type, Event Type pair indicating the nature of the message.
IO	Indicates whether the message is incoming, 'IN', or Outgoing, 'OT".
Sndg Apl	Application that is sending the message.
Rcvr Apl	Application to which the message is directed.
HDR	Message header. Use the right arrow key to view continuation of header.

You can use the right arrow key to view the continuation of the header (the full header is available if you scroll right):

```
MESSAGE ID #         Dte/TimeEntr'd Log Link   Msg:Evn IO Sndg Apl Rcvr Apl HDR
R RG CIRN  MSH^~&\^1600^MPI-STARTUP^NXT^RG CIRN^19980716075643^^ADT~A01^4^P^2.2^
  MPI-STAR MSH^~&\^NXT^RG CIRN^1600^MPI STARTUP^19980716075750^^ACK~A01^4^P^2.2^
R RG CIRN  MSH^~&\^1600^MPI-STARTUP^NXT^RG CIRN^19980716075643^^ADT~A01^4^P^2.2^
  MPI-STAR MSH^~&\^NXT^RG CIRN^1600^MPI STARTUP^19980716075750^^ACK~A01^4^P^2.2^




Col>  66  |<PF1>H=Help <PF1>E=Exit| Line>      4 of 4      Screen>   1 of 1
```

You can use all of the standard features of the VA FileMan Browser to search through the list. Some particular features of use are:

Keystroke	Description
Left and Right Arrow Keys	View text that is beyond either screen margin.
<PF1> Arrow Down	Page Down
<PF1> Arrow Up	Page Up
<PF1>F	Find Text
<PF1>N	Find Next

Documentation on the Browser is provided in the VA FileMan User Manual, and in the Browser's online help (in the Browser, press <PF1>H).

4.4. Viewing Message Text

While in the View Transmission Log option's main screen, you may want to view the full message text for a particular message.

To view message text:

1. Press <PF1>Z while viewing the transmission log.
2. At the "Enter Message ID Number:" prompt, enter the Message ID #. Note: Message IDs are unique for TCP/IP messages.
3. The Browser displays the requested message. To view characters beyond the right or left margins, use the arrow keys.
4. To return to the View Transmission Log main screen, enter R (for return).

For example, while viewing the transmission log, to view the message text for Message ID # 20 in the HL7 Message Text file, press <PF1>Z and enter the following:

```
Enter Message ID #: 20 <ret>
```

You would then see the text for that message in the new Browser window:

```
          HL7 MESSAGE TEXT:[JUL 16,1998@07:57:46] (wp): MESSAGE TEXT
MSA^AA^1



Col>   1  |<PF1>H=Help <PF1>E=Exit| Line>      2 of 2      Screen>   1 of 1
```

To return to the View Transmission Log main screen, enter R (for return).

5. Exception Processing API

The V*ISTA* HL7 package does not currently provide an HL7 exception log, although eventually it will. In the meantime, for packages maintaining their own exception logs, the HL7 package provides an API to enable reprocessing messages that you've logged as exceptions.

5.1. Why Support Only TCP/IP Interfaces?

As originally released in patch HL*1.6*36, the Exception Processing API worked for HL7 messages sent over existing transport layers (VA MailMan, DHCP-to-DHCP and serial). As modified by patch HL*1.6*19, however, these entry points, for the time being, only work with messages sent over TCP/IP logical links.

The reason for this is that the HL7 package is in the midst of a file conversion. Because enhancements are being released through patches, for the time being only messages sent over TCP/IP use the new file structures. In the future, another patch will convert all remaining messages to use the new file structure. At that point (as with the View Transmission Log option) all messages will be supported by the Exception Processing API entry points.

5.2. How to Handle Exceptions

An example of an exception is when an incoming message references an entry in a table that is missing. The message must either be rejected or stored as an exception for later processing.

To deal with exceptions, your message processing routine should:

1. Store the IEN of the message with a problem in your local exception log. You can get the IEN from the variable HLMTIENS, which is always defined during message processing.
2. Call $$DONTPURG^HLUTIL or $$SETPURG^HLUTIL(1) (they're equivalent calls) to prevent the HL PURGE TRANSMISSIONS option from purging the message.
3. Fix the error condition that is causing the exception.
4. Call $$REPROC^HLUTIL to reprocess the message.
5. If the reprocessing is successful, call $$TOPURG^HLUTIL or $$SETPURG^HLUTIL(0) (they're equivalent calls) to once again allow purging of the message by the HL7 package.

5.3. $$DONTPURG^HLUTIL

This entry point sets a message's DON'T PURGE flag (#773.10) to prevent a message from being purged by the HL PURGE TRANSMISSIONS option. As of patch HL*1.6*19, use only for messages sent over TCP/IP links (you must know this in advance).

$$DONTPURG^HLUTIL, $$TOPURG^HLUTIL and $$SETPURG^HLUTIL should be called only by message processing routines invoked by the HL7 package:

- The PROCESS ROUTINE of an interface
- The PROCESS ACKNOWLEDGEMENT ROUTINE of an interface
- The routine passed as a parameter to $$REPROC^HLUTIL(IEN,routine)

format	$$DONTPURG^HLUTIL()	
input	HLMTIENS	IEN of message in File #773. This variable should already be defined if in a message processing context.
output	return value	1: The message's purge flag has been set. -1: The call has failed; nothing has been changed.

5.4. $$TOPURG^HLUTIL

$$TOPURG^HLUTIL clears a message's DON'T PURGE flag (#773.10), allowing the message to be purged by the HL PURGE TRANSMISSIONS option. As of patch HL*1.6*19, use only for messages sent over TCP/IP links (you must know this in advance).

$$DONTPURG^HLUTIL, $$TOPURG^HLUTIL and $$SETPURG^HLUTIL should be called only by message processing routines invoked by the HL7 package:

- The PROCESS ROUTINE of an interface
- The PROCESS ACKNOWLEDGEMENT ROUTINE of an interface
- The routine passed as a parameter to $$REPROC^HLUTIL(IEN,routine)

format	$$TOPURG^HLUTIL()	
input	HLMTIENS	IEN of message in File #773. This variable should already be defined if in a message processing context.
output	return value	0: The message's purge flag has been cleared. -1: The call has failed; nothing has been changed.

5.5. $$SETPURG^HLUTIL

$$SETPURG^HLUTIL is an alternate entry point that duplicates the functionality of both $$TOPURG^HLUTIL and $$DONTPURG^HLUTIL.

format	$$SETPURG^HLUTIL(status)	
input	status	1: Sets the message's purge flag such that the message won't be purged (equivalent to $$DONTPURG^HLUTIL). 0: Sets the message's purge flag such that the message will be purged (equivalent to $$TOPURG^HLUTIL).
	HLMTIENS	IEN of message in File #773. This variable should already be defined if in a message processing context.
output	return value	0: The message's purge flag has been set or cleared. -1: The call has failed; nothing has been changed.

5.6. $$REPROC^HLUTIL

This entry point lets you *reprocess* a message. Pass the routine to reprocess the message as a parameter to the API. Reprocessing is essentially the same as when, in an interface, the HL7 package calls an application's PROCESS ROUTINE (for incoming messages) or PROCESS ACKNOWLEDGEMENT routine (for incoming acknowledgements). As of patch HL*1.6*19, use only for messages sent over TCP/IP links (you must know this in advance).

format	$$REPROC^HLUTIL(IEN,routine)	
input	IEN	IEN of the message in file #773 to reprocess.
	routine	The routine to Xecute to reprocess the message.
output	return value	0 means call has been successfully completed. -1 means call has failed; nothing has been changed.

6. TCP/IP LLP Implementation Details

The information in this chapter is useful if you are interfacing a device or system with **V*IST*A** over TCP/IP and need to know the message format expected by **V*IST*A**.

The TCP/IP implementation in patch HL7*1.6*19 encapsulates HL7 messages using Minimal Lower Layer Protocol (MLLP), rather than Hybrid Lower Layer Protocol (HLLP). MLLP is a very simple protocol, and can be used because the TCP/IP channel itself provides most services needed for error-free transmission of messages, including:

- Connection Handshaking
- Full Duplex Data Transfer
- Error Detection and Retransmission
- Flow Control
- Congestion Control
- Connection Release

Using a particular lower layer protocol is not required by the HL7 Standard but is highly recommended.

MLLP Message Format

The MLLP message format expected by **V*IST*A** is as follows:

- Start Block = $C(11)
- End Block = $C(28)
- Carriage Return = $C(13)