InfiniLinks Version 2.0.0

Release Notes

Dagran Healthcare Systems Consulting

Thank you for using InfiniLinks. The version 2.0 release of the InfiniLinks software has no restrictions on use and may be installed in production environments at each customer's discretion. This release features completely redesigned queues, a new server configuration utility, and significant upgrades to the Monitor and other command-line tools.

Installation Notes

Similar to previous versions of the program, the software is released as either two InstallShield installation packages (Windows), or in two compressed archive files (UNIX). The InstallShield packages are separated into server and client application packages. The server package includes the InfiniLinks server and compiler tools, while the client package includes the Monitor and Remote Configuration Applet, a tool to configure the server initialization file. The two files under UNIX include a binary archive, which includes all InfiniLinks executable and library files, and a configuration archive, which contains updated configuration files.

Under Windows, any previous version of InfiniLinks must be manually uninstalled prior to installing this software. A bug in the installation routines for versions 1.3 and higher necessitates this action. Failure to uninstall prior versions of InfiniLinks will result in multiple entries for the InfiniLinks Server and Monitor installation packages within the "Add or Remove Programs" window in the Control Panel. The software will work properly however. Manually uninstalling the software is only required once. Future releases will correctly install over previous versions.

Due to a complete rewrite of our queuing software, queues are not compatible between previous releases and this version of the software. If performing an upgrade, it is highly recommended that the contents of the queue directory be completely deleted prior to starting the server install process.

Version 2.0 introduces many new changes to the Server Initialization File (SIF), located in the */config* directory underneath the root installation directory. To aid you in migration your configuration file to version 2.0, a utility called *convertcnfg* has been included. The utility will update the SIF and all communications component configuration files to meet version 2.0 requirements. The utility will also backup and remove your version 1.4 queue files, as they are incompatible with version 2.0.

There have been some additions to the default InfiniLinks directory structure. The following directories have been added to the base InfiniLinks directory:

Directory	**Description**
My Profiles	This directory is the default storage location for all Monitor profiles.
Tbl	The *tbl* directory is the default storage location for table conversion files. The RCA will create files in this location by default.
temp	The *temp* directory is used to store temporary files used to automatically rebuild the queue files in the unlikely event that the server experiences a hard crash.

The following files have been made obsolete in this release and should be manually removed from the *bin* directory when performing an upgrade:

File Name	Description
dhctapi.dll	This file, which contained DHSC's lengthy wrapper around the Windows telephony architecture, is no longer needed. The TAPI wrapper has been moved into *dhccdl.dll*.

The following binary files have been added in this release:

File Name	Description
ctreestd.dll	This file contains the core routines and implementation for our new queue files.
dhcbtree.dll	This file contains the software that implements an object-oriented wrapper around the API for the queuing software
dhccdlhtml.dll	This DLL is a resource DLL that contains the HTML templates used by the Remote Configuration Applet
dhcconfig.dll	This DLL contains the core routines used to build the communications client configuration files at run time.
dhclkup.dll	This DLL contains the core routines used to build and use the lookup tables added to the MML module
dhclog.dll	This DLL contains the upgrades and enhancements to the log file implemented in the server and in the monitor
dhcprof.dll	Profiles are no longer exclusive to the monitor. Several new utilities were added that use server profiles; hence the code used in the profiles has been moved into this DLL
qcnv.exe	This utility is used to modify the data in a queue after the queue's structure has been modified using the RCA or another tool
rca.exe	The Remote Configuration Applet. Use the RCA to modify the contents of a *server.ini* file without having to edit the configuration file directly. This is a cool little utility.

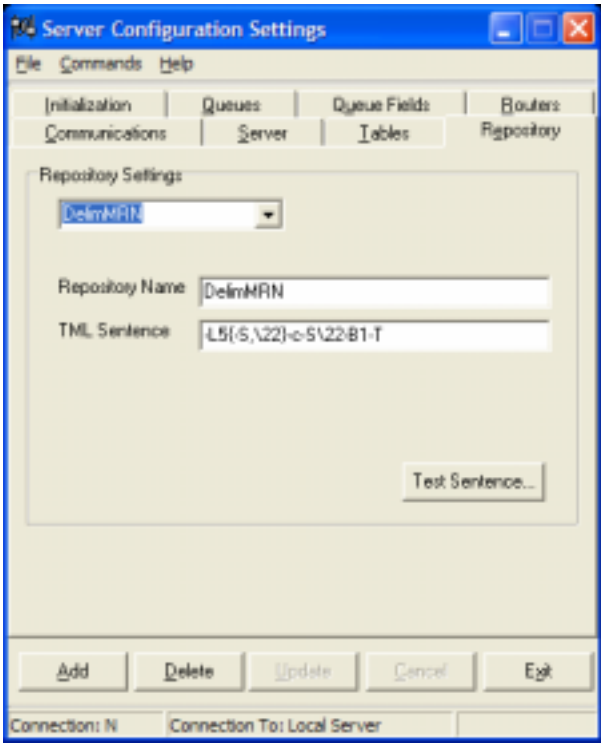
Removed Features

One feature that was introduced in version 1.4 of the server did not make it into release 2.0 of InfiniLinks. The ability to set the maximum size of a log file and archive log files once they hit a certain size is not available. These features will be included in a future release. The *MaxFileSize* and *Overwrite* parameters should be moved from the [Server] section of the SIF and placed in the [LogFile] section. Although these parameters will have no effect at this time, the parameters will be used when the feature is re-enabled in a future release.

Enhancements

TML Sub-module

TML Repository – A repository has been added where commonly used TML sentences can be stored and reused globally throughout the server. Use the RCA utility to add, modify, or remove entries within the repository. Use the new "-U" keyword to access an entry within the repository from TML.



In the example above, the *DelimMRN* TML Sentence could be used in the following TML sentence, "*-UDelimMRN*". Other TML functions may precede or follow the "-U" function, *"-m4-UDelimMRN-B4"*. Alternately, the –U function can be used multiple times within the same TML sentence, *"-UDelimMRN-UHL7Mrn"*. In the last example, the *HL7Mrn* Super Sentence would be executed immediately after the *DelimMRN* function completed.

TML has been modified so that the contents of a MML or CDL variable can be inserted into a sentence, using the new Variable function (-V). Given the sentence: *"-m5-Vx"*, where the source of the sentence is "HELLO" and the contents of *x* is "GOODBYE", the result of this sentence would be "HELLOGOODBYE".

InterpTree Application

XML support – Support for XML has been expanded within the utility. You can now add instances of a node and new child values by right-clicking on a specific node.

Printing – You may now print out the contents of a parsed message to a specific printer. You have the option of printing all nodes, or suppressing NULL fields.

MML Sub-module

Lookup Tables –Support for lookup tables has been added to MML. Below is a sample table:

```
"HL7Value"  "OutValue"
"(default)" "Unknown"
"M"   "Male"
"F"   "Female"
```

Each lookup table is divided into columns, where the leftmost column is the key value and the remaining columns contain the translated values. Each column is separated by a TAB character, and each value should be within quotes. A table can be built using the RCA as well:

Right click within the grid area to find the options for adding a row or column to the table. Duplicate entries within a table are not allowed. In addition to manipulating the table itself, the table file also allows you to configure the table within the server environment. Tables are stored within a section in the *server.ini* file. An example is provided below:

```
[Tables]
TableCount=1
Table0=Gender,c:\program files\dhsc\infinilinks\tbl\sex.tbl
```

Anyone familiar with previous versions of InfiniLinks should feel right at home with this entry. Within the [Tables] section, the *TableCount* value indicates the number of tables configured within the system. Each table entry is delimited by commas and named *Table0, Table1*, etc. The first element within a table entry is the name of the table and the second entry contains the path and file name of the table. Table names are case sensitive and are global to the server, in that any MML file within the server may use the table.

The syntax for using a table within an MML script is as follows:

```
Lookup <source> to <target> using <literal> column <literal>
```

The source argument is the token within a message that contains the value you would like to replace. The target is the node in the outbound message where the corresponding value will be written to. The literal value after the using clause is the name of the table and the literal following the column keyword is the column which contains the target value.

Using the example table file above, we could use the following code in an MML script:

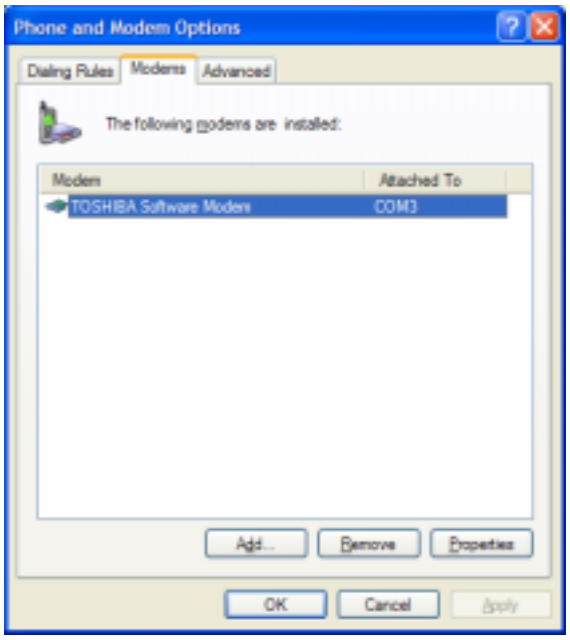
```
lookup "%A01_PIDSEGMENT.sex" to "%A01_PIDSEGMENT.sex"
using "Gender" column "OutValue"
```

So, in the example above, if the sex field of the PID segment contained "M", the value "Male" would be written to the outbound message. If the inbound field contains an "O", which is a value not in the table, the default value would be used, and "Unknown" would be written to the outbound message.

CDL Sub-module

Serial Protocol Support – Yet another communications protocol has been added to the CDL sub-module. In addition to using TCP/IP, ODBC, and file-based processors, you can now use a serial device to transfer data from one endpoint to another. Below you will find a list of the parameters required to set up the [comm.] section of a communications client initialization file:

Device – The 'Device' parameter is used to specify the TAPI device name for the modem or TAPI-compliant communications device. The name for a particular device can be found by clicking on the "Phone and Modem Options" icon in the control panel. From the example below, the value for the 'Device' entry in the communications client configuration file would be 'TOSHIBA Software Modem'. Please be advised that the value is case sensitive.



ConnectType – The value for this parameter must be either 'Host' or 'Dial'. By setting the value to 'Host', the TAPI device will be set to answer any incoming calls. By setting the value to 'Dial", the communications client will dial the number specified in the DialString parameter specified below.

DialString – The 'DialString' parameter is used in conjunction with the 'Dial" value in the 'ConnectType' parameter. When set to dial out, the communications client will pass the value entered to the TAPI device to dial out. In other words, this is where you would enter the phone number if making a call to another modem across a phone line.

Queue Sub-system

As previously mentioned, the queue files have been completely reworked for the release of the server. First of all, the queue files now support journaling and automatic recovery, which means that a queue will automatically rebuild itself if it becomes corrupted. Queues now support field values, which allow you to extract values from a message and store it in its own field to quickly search on the message. Finally, queues can now be configured to retain messages after they have been processed. Each queue can be configured to retain up to a maximum of 50,000 messages before beginning to delete them. Each queue will default to retaining a thousand messages.

All of these enhancements necessitate big changes to how queues are configured within the *server.ini* file. In addition to the [queues] section, each queue now maintains its own section within the *server.ini* file. An example is provided below:

```
[Queues]
QueueCount=3
Queue0=SampleQueueIn,C:\InfiniLinks\Queues\SampleQueueIn,1000,I,5000,START
Queue1=SampleQueueOut,C:\InfiniLinks\Queues\SampleQueueOut,1000,O,50,START
Queue2=TestMemory,C:\InfiniLinks\queues\TestMemory,1000,I,10000,START
```

Similar to the [Tables] section, if you have worked with previous versions of InfiniLinks in the past then you should feel right at home. In the [Queues] section, the *QueueCount* parameter indicates the number of queues the server has configured. Each subsequent entry in the list should be named *Queue0, Queue1,* etc. The value for each entry is a series of arguments delimited by commas that describe the characteristics of the queue. The arguments are as follows:

QueueName **–** The name of the queue.

Queue File **–** The path and file name of the queue. *Do not specify an extension to the file name!* Based on the file name and path provided, the server will actually create several files in that directory.

Sleep Interval **–** This value is no longer used, but is retained for backward compatibility

Queue Direction **–** This value indicates whether the object is an inbound queue or an outbound queue. Possible value are “I” and “O”.

Max Messages **–** This value indicates the numbers of messages to retain in the queue before the messages are overwritten. Each queue can retain a maximum of 50,000 messages. Please note that due to queue structure changes, queues take up significantly less disk space than they have in the past.

Default State **–** The default starting state for the queue. See the server enhancements section for more information.

In addition to retaining messages in the queue, each queue can be configured to extract specific values from each message and store them in a separate field in the raw message. This allows the end user to quickly see what patient a message applies to or quickly search a field for a specific value. An example of the fields section for a specific queue is listed below:

```
[SampleQueueIn]
```

```
FieldCount=2
Field0=LastName,10,Hl7LastName,1
Field1=BirthDate,10,HL7BirthDate,0
```

As you can see, the section name is the name of the queue itself. The first entry is the *FieldCount* entry, which indicates the number of fields that will be associated with the queue. The next entry should be named *Field0*, the following entry *Field*, and so on until the number of entries matches the value specified in *FieldCount.* The format for each field entry is specified below:

Field Name – The name of the field. Field Name is case sensitive and each field name must be unique for the queue it is associated with.

Field Length – The maximum length of the field. A field length can range from 0 to 100 bytes in length. If the value for a given field in a record is greater than the maximum length, the value will be truncated to fit the field.

TML Super Sentence – The contents of each field are extracted from the raw message passed to the queue by executing a TML sentence against the raw message. The result of the sentence is the value that will be placed in the field. The value for this entry must be the name of an entry in the TML repository. A TML Sentence cannot be hard-coded into this field.

Index Flag – This argument indicates whether the field should be indexed within the queue. A value of "0" indicates that the field should not be indexed, while a value of 1 indicates that the field is indexed. A field that is indexed can be searched on within the monitor. In the example for Field0 above, the Last Name field is marked as indexed, which means that the end-user can use the Monitor to search for all records in the queue where the last name equals "ROGERS."

Once a queue has been created and fields have been associated with it, the structure of a queue may not be changed simply by changing the field values in the *server.ini*. A special utility called *qcnv.exe* must be run to convert the existing data in the queue to its new format. The syntax for the command is *qcnv <queue-name>*.

The server's log file will indicate when a queue's structure has changed and that *qcnv* will be needed to run against the file.

InfiniLinks Server

Reload MML Files – It is now possible to make a change to an MML or MDX file and apply the changes without needing to reboot the server. From the Monitor, right-click on the Router, and click "Reload MMX File".

Default Object State – The server has been reworked so that each high level server object can be set to a specific state when the object is started. Queues and routers can be set to be either stopped, started, or suspended at start up. A communications client can be set to either stopped or started (although it may appear suspended in the Monitor if it cannot connect to its endpoint application).

This setting can be configured for each object using the *RCA.* The values can also be modified by editing the *server.ini* file directly. For each object, the last parameter is the object's default state. The following values are supported: STOP, START, SUSPEND.

Monitor

First off, we have completely rewritten the code behind the generating of the Monitor HTML pages. This enhancement allows the programmer to implement new HTML pages efficiently and in a timely manner, many of the HTML pages have been revamped. Other pages had pieces added and other pages had pieces removed which are no longer being used, such as the "DebugPort".

Listed below are the major enhancements to the Monitor:

- ⇒ New Server Explorer Nodes
  - Server Log
  - Error Queue
  - Messages
- ⇒ Menu Options
  - Searching Log File
  - Searching Queues
- ⇒ Function Keys

New nodes have been added to the Server Explorer: Server Log, Error Queue, Messages. The Server Log and Error Queue node can be found by expanding the Connection Name node, which is the "Default (Testing Local Profile)" in the screen shot. The Message nodes are tied to their own individual queue. For reference please see the screen shot below:

The user may now browse the log file from within the Monitor. The Server Log node is used to display the dhc.log file. When viewing the log file, the server will push updates to the Monitor, allowing automatic updates to be displayed. The log file display is configurable in the server initialization file:

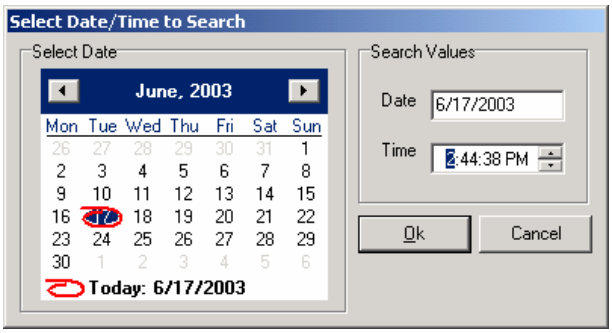
```
[LogFile]
MaxPages=128
LogFile=C:\Program Files\DHSC\InfiniLinks\Log\dhc.log
MaxPageSize=5
```

MaxPages – This element defines the maximum number of pages to be held by the server.

LogFile – This is the path and name of the log file used by the InfiniLinks Server ©. Anyone familiar with the InfiniLinks product will be familiar with the dhc.log.

MaxPageSize – This element defines the size of each page for the log file in kilobytes. For the example above the page size is 5k.

The log file is fully searchable by date. This function is accessible by right clicking the Server Log node. The following window is then displayed:



The hour, minute, and second field can be incremented with the up/down arrows, as the Windows® Date/Time Properties window. Simply select the field to increment or decrement, and press the corresponding arrow. After selecting search values and pressing Ok, the log file will be displayed at the first occurrence of the selected values. Also, this option is **only** available when the Server Log node is selected.

More robust error handling options have been included in this release. There is now an error queue which holds messages that have generated translation errors in the router. The Error Queue Node displays the errored messages. In the event an error occurs during the translation, the record is sent to the error queue from the router. The total number of messages to be displayed per page can be configured in the Option menu item.

The HTML page for the error queue will supply the administrator with each of the following listed:

Field Name	Description
Index ID	The index number is a unique number assigned to every record sent to the error queue.
Queue	The Queue field lists the name of the originating queue where the message was stored.
Transform ID	MML files may have multiple transformation-objects. This ID indicates as to which transform statement the message generated an error. Value of -1

	indicates the message failed identification across all objects.
Error Code	The error code indicates to the user with the number associated with an error in the MML section of the strings.ini. Clicking the error code will supply a detailed description as to why the message was placed in the error code.
Message Status	This field is used to display the status of the message. A status of "ERROR" signifies the message has not been reprocessed. A status of "REPROCESSED" means the message was corrected and re-transmitted.

Clicking the Index ID number of the message will display the error message in a raw format. Once viewing the message detail you have some options to choose from:

- Edit the raw message in the text area.
- View the message in the Interp Tree. This may give the user more information as to why the message errored out.
- Delete the message from the Error Queue.
- Update the message if any changes were made.
- Use the "Reprocess" button to send the message to the router for reprocessing.

All of the above options are available on the message detail page as buttons.

Every message received by the queue is stored in a database, and has both fixed and user defined fields. Selecting the Message node will display the message stored within a particular queue. The Monitor will display a set number of messages per page, which is determined by a value set by the user using the options menu. The default value is 25.

Fixed Fields	There are three fixed fields associated with to each message. They are: Index ID, time stamp, and the message status. These fields are inserted into the queue automatically and are always displayed.
User Defined Fields	Users can specify any number of fields along with the fixed fields. Each queue is assigned there own user defined fields within the server initialization file. The data, from the unique fields, is copied from the raw message using a super sentence in the TML Repository. These values can be indexed for quick searches. See the Queues section for more information.

As a queue is populated, it will obviously soar above the 25 messages to be displayed per page. Therefore, located at the bottom of the list page is two buttons, labeled "Prev Page" and "Next Page". Pressing either button will either display the previous 25 records or the next 25 records in the queue.

Selecting an Index ID from the Message list screen will display the message detail. This page displays the record in a raw text format, along with the field names and values for each field. From this page the user can view the record using Interp Tree, requeue the message, reindex, delete, and even update the message.

The following table is a breakdown for each button available on the message detail page:

Button Title	**Action performed**
Tree View	The Tree View button is responsible for launching the Interp

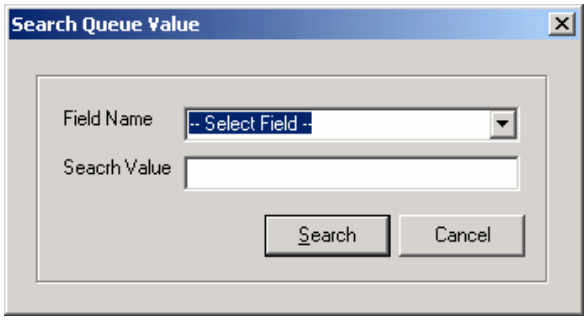
	Tree. This application allows the user to view the message in tree view parsed out format. Nodes can be edited within the Interp tree, and saved back to Message Detail Page.
Requeue	This button will create a copy of the record and insert it into the queue to be reprocessed. It has no effect on the original message.
Reindex	Reindexing allows the user to reprocess a range of messages. For example, if there are 10 records in the queue with ID's of 1-10, and ID 5 is selected. Pressing reindex will reprocess ID's 5-10.
Delete	Will flag the record for deletion. The record is deleted when the queue reaches its maximum number of messages.
Update Msg	Updates the message in the database to reflect any changes made to the raw text message.
List Page	Returns the user to the Message List Screen.
Prev Record	Displays the previous Index ID detail page.
Next Record	Displays the next Index ID detail page.

Any changes made to a message in Tree View will automatically update the message on the Message Detail Screen and in the database.

Messages stored in queues are broken into fields and has three common fields for every message:

1. Index Id
2. Time Stamp
3. Status of the message

Users can define unique fields for each queue that can be indexed as well. The unique fields can be defined within the Server Initialization file and there is one section for each queue, please refer to the Queues section for further details. Defining unique fields gives the user the power to search pre-defined unique fields. When selecting the Search Queue option from the context menu a dialog box is displayed as the one below.



Steps for searching the queue are as follows:

1.	Select a field from the drop down box. Only indexed user defined fields are searchable.

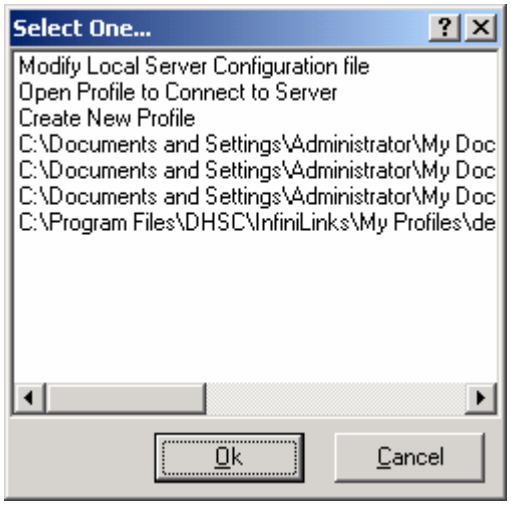
2.	Enter a value to search for in the "Search Value" text box.
3.	Select Search.
4.	The results will be returned in a search result page. From this point the page is similar to the message list and error list pages.

The Monitor is now loaded with functions keys and shortcuts. For a complete list see below:

Function	**Description**
F1	Context sensitive help.
F2	Stop selected component (Router, Queue, Client)
F3	Start selected component (Router, Queue, Client)
F4	Suspend selected component (Router, Queue)
F5	Refresh
F6	Hook
F7	Unhook
F8	Send Message to selected queue
F9	Display configurable options
F10	Connect profile
F11	Expand Server Explorer
F12	Collapse Server Explorer
Shortcut	**Description**
Ctrl + N	Create new profile
Ctrl + O	Open and connect profile
Ctrl + E	Edit Profile
Ctrl + D	Disconnect selected profile
Ctrl + X	Exit Monitor (Alt + F4)
Alt + 1	Open the first most recently used profile
Alt + 2	Open the second most recently used profile
Alt + 3	Open the third most recently used profile
Alt + 4	Open the fourth most recently used profile

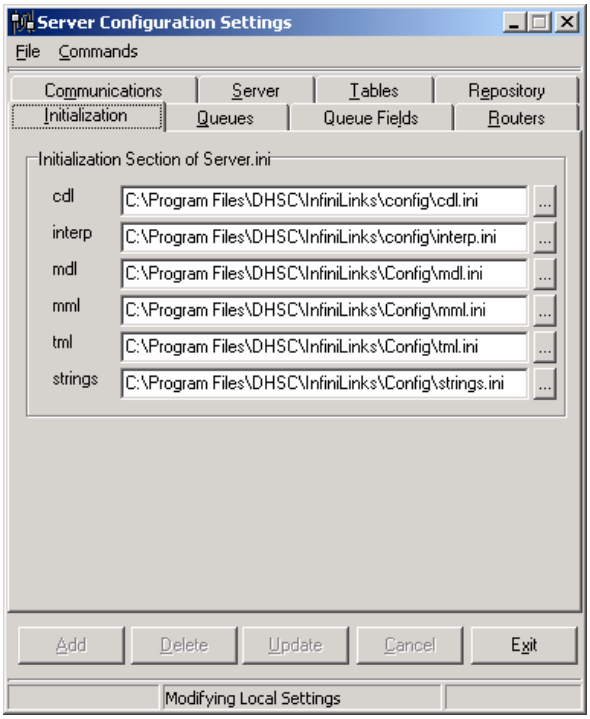
Remote Control Application

The Remote Control Applet (RCA) is used for configuring your server initialization file in a user friendly environment. It can modify the server file in one of two ways. The first is to connect locally and modify the necessary files directly. This type of connection does not require the InfiniLinks Sever © to be in an active state. The second type of connection is a remote connection through a network using a DHC profile, the same profiles used by the Monitor. The screen shot below shows an example of the connection window used by the RCA. Selecting a profile and holding the mouse over it will display a help box to let you know what file you have selected, otherwise the user can scroll right.



A remote connection downloads all necessary files fro the server and stores the data in temporary files which the RCA will use to simulate a local connection. Before existing, all data is sent back to the InfiniLinks Server © and updated on the remote server. The save request can be found on the main menu under the Commands option. Once saved, the server will need to be stopped and restarted for the changes to take effect.

After selecting the type of connection to establish, the RCA application is then opened. The RCA is laid out with multiple tabs. Each tab represents a section from the server file, so if queues need to be added or updated select the Queues tab or if the server port needs to be changed select the Server tab. The screen shot below shows an example of the RCA:



Notice the buttons located at the bottom. These buttons are universal and used for every tab. The table below explains the functionality for the buttons:

Button	Description
Add	The Add button is used for adding new elements to one of the following: Queues, QueueFields, Routers, Communications, Table, and Repository. Selecting this button will automatically update the section count, and name the next element appropriately. This button does not apply to the Initialization and Server tabs.
Delete	Delete will delete an element from a section and rename all the other elements accordingly. This button is not available on the Initialization and Server tabs.
Update	Update is enabled when a change is made to an existing element on any tab. Selecting update will update the server initialization file.
Cancel	This button will cancel any changes made since the last update. However, changing between tabs and elements in the Queues, QueueFields, Routers, Communications, Table, and Repository will automatically update the server initialization file.
Exit	This button will exit the RCA application. If a remote connection is established and changes have been made, the RCA will prompt the user to save changes back to the remote server.

Most of the information within the tabs will be familiar to users of the current release of InfiniLinks. More detailed documentation will be provided at a later date.

Documentation

We are pleased to announce that new documentation and help files have been included with this release of the software. Documentation for the Monitor and RCA is provided in the form of Help files, which are located in the */help* directory underneath the default installation directory. The help file for the Monitor is *monitor.hlp*, while the help file for the RCA is, oddly enough, *rca.hlp*.

The User's Guide has been completely revamped for version 2.0. The guide now includes full references for the MDL, MML, and CDL languages, as well as comprehensive tutorials for maintaining queues and administering the server. The reference guide can be found in the *doc* directory under the InfiniLinks root directory.

Known Issues

InfiniLinks Server

- **Duplicate error messages in log file (00001167) –** The InfiniLinks server will occasionally send two or more copies of the same error message to the log file if a configuration error within the [Queues] section of the SIF prevents the server from starting. This bug has no impact on server operation.
- **Queue Manager will not create a queue with zero field definitions if the repository is not configured (00001168)** – The Queue Manager will not create a queue, even if no field definitions are configured, if the [Repository] section of the TML configuration file is not present. The default behavior of the Queue

Manager should allow a queue with no fields to be created regardless of the status of the TML Repository.

- **Resource leak**: **log file object occasionally not deleted if a configuration error prevents the server from starting (00001170) –** Under certain circumstances, the server will occasionally not free the memory associated with the log file object if a certain set of configuration errors occur. This bug has a negligible impact on performance, unless the server is continually stopped and started without ending the *dhcserv.exe* application.
- **Some configuration file changes not applied without exiting *dhcserv.exe* process (00001170) –** If using *dhcserv.exe* under Windows, changing a value in a configuration file other than the server.ini file requires the *dhcserv.exe* process to be terminated and restarted before the changes will be applied. Stopping and starting the server within the DHC Server Manager application has no effect.
- **Starting two instances of the server corrupts queue files (UNIX only) (00001160) –** If two instances of the DHC server running under UNIX attempt to access the same physical files for a queue, the queue will become corrupted.

Monitor

- **List Page button does not work properly in Search Detail Page (0001114) –** When viewing the message details page from the search results window, clicking on the 'List Page' button returns the user to the 'Queue List' page, rather than the 'Search List' page. To go back to the Search List page, right-click on the HTML page to bring up the Internet Explorer context menu. Click on 'Back.' This should return you to the 'Search List' page.
- **Search Queue function appears to hang after entering search criteria (0001110) --** The queue search function in the Monitor does not place any limits on the number of records that the server may return. If your search criteria returns a large number of records (200+), all records will be returned from the server at once. While processing these records, the Monitor will appear to hang, although it will eventually return control to the end-user. The only workaround at this time is to narrow the search criteria to limit the number of records returned.

Remote Control Application

- **Missing communications component initialization file prevents save (00001170) –** If you place a missing file name for the communications client configuration file parameter, you can set up your configuration file, but the RCA will never save the file to disk. To work around this, leave the initialization file name parameter blank. The RCA will prompt you for a file name at the appropriate time.

TML Sub-module

- **TML sentence enters endless loop when last command in loop fails (00001068)** – When putting together a loop until fail construct, if the loop command fails on the last TML function within the loop, the sentence will not break out of the loop and will enter an endless loop. The following sentence will cause the TML engine to enter an endless loop: -f-L{-A-c-m1}. The move 1 command will fail when the end of the string is reached. However, the loop does not detect the fail condition and continues to execute the loop.

CDL Sub-module

- **Communications component locks up when logging messages to server log file (00001161) (00001110) --** Communications components will consistently lock up when writing large amounts of data to the *server* log file. If you need to log large quantities of information