MO71: WebSphere MQ for Windows GUI Administrator User Guide Version 5.3.2

2nd February 2003

Paul Clarke
WebSphere MQ Development
MP211,
IBM UK Laboratories Ltd.
Hursley
Winchester
Hants, SO21 2JN
United Kingdom

paulg_clarke@uk.ibm.com

Take Note!

Before using this User's Guide and the product it supports, be sure to read the general information under "Notices".

Tenth Edition, February 2003

This edition applies to Version 5.3.2 of *WebSphere MQ for Windows GUI Administrator* and to all subsequent releases and modifications until otherwise indicated in new editions.

Notices

Contents

Notices ii

Preface vi

Main changes from previous version vii

Chapter 1. WebSphere MQ GUI Administrator 1

Overview 1

Installation 2

Chapter 2. Getting Started 3

Running the Administrator 3

Adding a Queue Manager Entry 4

Connecting via a Client 4

Connecting via a different Queue Manager 4

Chapter 3. Issuing Commands 5

The Command window 5

Fields 5

Status window 6

Action buttons 6

List Windows 6

Examples 7

Display a list of Queues 7

Defining a Queue 7

Defining a Queue like another Queue 8

Starting a Channel 8

The Pop-up menu 9

Alter list dialog 10

MQSC Window 11

Examples 12

Chapter 4. Filtering 13

Examples 16

Chapter 5. Queue Manipulation 18

Fields 20

Display Range 20

Max Message Size 20

Target Queue Manager 20

Target Queue 21

Pop-up Menu 21

Next 21

Prev 21

Display Format 21

Detail Level 22

Copy To Clipboard 22

Copy 22

Move 22

Delete 22

Convert 22
Strip Headers 22
Multi Transaction 23
Message Selection 23
Context 23

Chapter 6. Network View 24
Network Display 24
Display Format 25
Verify Display 26
Queue Managers Display 27
Levels Display 28
Problem Selection 28
Network Names 29

Chapter 7. General Actions 30
Printing 30
Printing Lists 31
Exporting Definitions 32
Export MQSC format 32

Chapter 8. Monitoring 33
Command Strings 33
Format 34
Substitution Characters 34
Examples 34

Chapter 9. Customization and Reference 35
Menu options 35
File 35
Commands 35
Action 36
View 36
Preferences Dialog 38
Options 38
Dialogs 40
Verify Network 40
Sounds 41
Display 41
Location Dialogs 41
Monitoring 43
Buttons 45
Container Windows 46
Keyboard Control 46
Reply Queue Details 46
Model Reply Queue 47
Colour Dialog 47
Extended Selection 47
Parameters 48

Chapter 10. Authorities 49

Chapter 10. Trouble Shooting 52

Chapter 11. Migration from previous versions 53

Migrating from version prior to Version 5.3.2 53

List Selection Colour 53

Client Configuration 53

Autorefresh resolution changed to 1 second 53

Change default reply queue 53

Migrating from version prior to Version 5.3 53

Migrating from version prior to Version 5.2 54

Options 54

Objects in the main window 54

List and Tree View 54

Migration from version prior to Version 5.0 54

Migration from a version prior to Version 4.0 54

Functionality 54

Main changes 55

Configuration 55

Appendix A: Icons 56

Table 1: Queue Manager Icons 56

Table 2: Queue Icons 56

Table 3: Channel Icons 56

Table 4: Other Object Icon 56

Table 5: Network View Icons 57

Table 6: Container Icons 57

Appendix B: Problem Selection List 58

Appendix C : Configuration file 59

Preface

This SupportPac was originally written as a monitoring program for the 1996 Olympic games in Atlanta. The requirement was for an OS/2 PM program which would actively monitor a number of remote Queue Managers and give visual and audible feedback if a problem was detected. At the time I was interested in gaining experience of PCF (Programmable Command Format) messages. Since the program already had the basic structure required I decided to add command support to it allowing you to issue virtually any MQSC command against any number of Queue Managers in a dialog.

The program was written largely in my own time and, as such, many facilities I would like to add have not yet been written. In addition it has not had the benefit of any formal testing. I can therefore not vouch for the correctness of the program and would welcome any comments, both positive and negative. Either way I hope you find the program useful and a big usability improvement over MQSC.

My thanks go to Morag Hughson for considerable assistance with this manual, for numerous usability suggestions, for keeping me sane when nothing ever seems to go right, as well as tirelessly testing many buggy versions of the program!

I would also like to express my gratitude to the many people over the years who have written to me to express their thanks for the program and make suggestions. It makes all the difference in the world to know that people are using the program and finding it useful. When I first wrote the program there were few alternatives to MQSC for Queue Manager administration but now there are many packages available. Although this is a spare time activity I shall try to ensure that updates are made to this program as long as people tell me it is useful to them.

.

Main changes from previous version

The main changes are :-

1. Cluster objects included in Network View.
2. Multiple Network views allowed.
   Each location can be *associated* with one or more network names
3. Dialog Windows need not be on the windows taskbar.
4. Export option available in text mode only on virtually all dialogs.
5. Option of having main application icon go red if any Queue Manager unavailable.
6. Ability to set application locale / codepage.
7. Queue / Message browser now understands RFH2 / XML format.
8. Existing dialog window, for example QM1 Queue list, can be displayed again if reselected rather than creating new window .
9. Main window and Network view now display processes and namelists.
10. List dialog selections can be coloured independently.
11. Colour scheme support.
12. List windows titles look and behave like buttons.
13. More configuration options, such as print settings, saved across restart.
14. Dialogs can be ended by <esc> key.
15. Default Maximum Browse Message size now configurable.

Chapter 1. WebSphere MQ GUI Administrator

This document describes the functions available in the SupportPac.

Overview

The main window displays a list of Queue Managers. Typically, this list would be the local Queue Manager and a number of other 'remote' Queue Managers that the machine has access to via WebSphere MQ channels. The user can then perform the following functions against the Queue Managers in the list :-

1. Issue commands
   By selecting a Queue Manager in the list and invoking one of the Command menu options the user can issue commands against that Queue Manager. Note that the Queue Manager can be any platform that has a command server (including z/OS).
   Commands may also be entered in MQSC format from the MQSC window described later.

2. Filtering
   When dealing with a list of objects as the result of an issued command, it can be useful to filter out some of these objects to reduce the size of the list displayed.

3. Queue Manipulation
   The queue being accessed can be either on the local Queue Manager or a remote Queue Manager provided a program like 'MQMONNTP' is running on the remote Queue Manager. Connecting to the Queue Manger as a client will allow queue manipulation on any remote Queue Manager since, as far as MQMONNTP is concerned, the Queue Manager is local.

4. Communication
   By selecting Talk from the menu, a dialog is shown which allows the user to type a text message that can be sent to the target Queue Manager. The receiving Queue Manager must be running the monitoring program, in which a dialog shows the text that the original Queue Manager sent.

5. Monitoring
   With monitoring enabled, the program will periodically send 'monitor' messages to any enabled Queue Manager. You can monitor a Queue Manager that the application is directly connected to but it is more useful to monitor a remote Queue Manager. This essentially involves sending a message to a named queue ‘the monitor queue’ on the remote machine and waiting for the message to be returned.

Installation

Create a directory, say MQMON, and copy the following files into it :-

- MQMONNTP.EXE

Note that the directory where you place the .EXE program is the default location where the program will store its configuration file MQMON.CFG and the object definition file MQMON.DFN. You generally need not worry about these files but you should be aware that they are written and therefore the program needs write access to the directory. Alternatively you can use the '-f' parameter to the program to give the directory name the program should use for these files.

Once the MQMONNTP program is in the path it can be run just by typing 'MQMONNTP' on the command line. No further set-up at this point is required. The program will dynamically load either the WebSphere MQ client or WebSphere MQ Queue Manager libraries as required. Note that to successfully connect to a Queue Manager at least one of these WebSphere MQ products must be installed. The windows WebSphere MQ client is available for free download at :-

http://www-3.ibm.com/software/ts/mqseries/txppacs/txpm2.html#cat3

Users will probably find it more convenient to set up a desktop icon for the Administrator program.

Chapter 2. Getting Started

This section will explain how to run the Administrator program and provide an example to add the first Queue Manager to be administered.

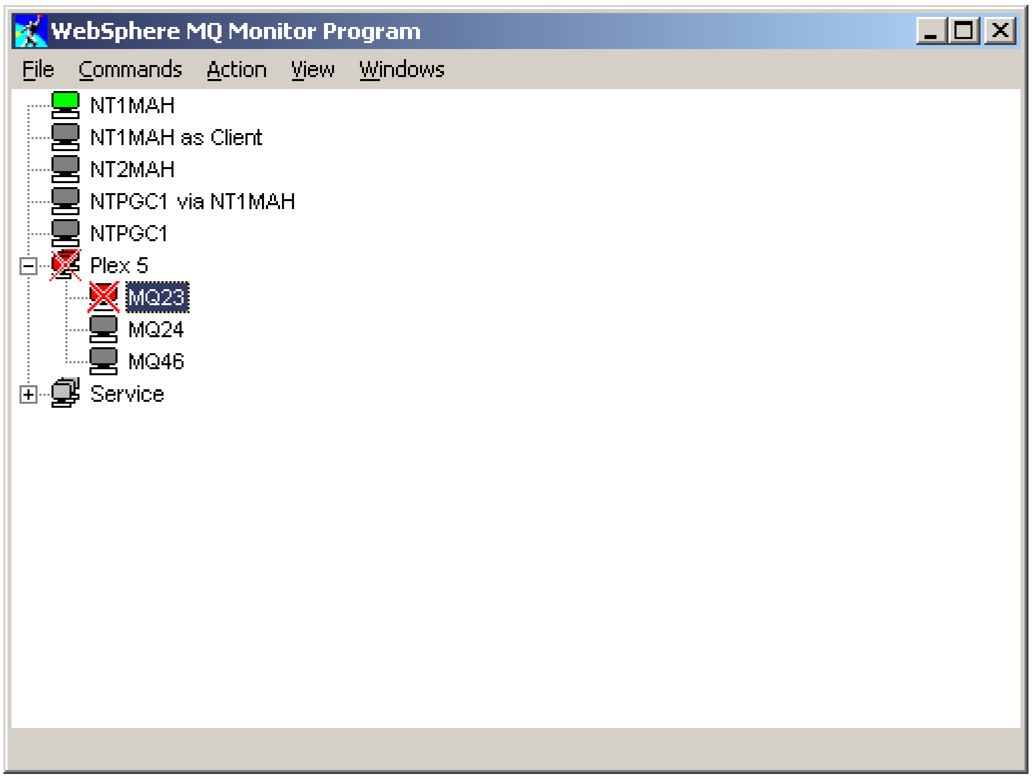
Running the Administrator

To run the Administrator enter the following :-

- *MQMONNTP*
- Or click on the desktop icon created at installation time

The program does not **require** any parameters but you can pass a Queue Manager location in on the command line if you wish. See *"Parameters"* on page 48 for more information.

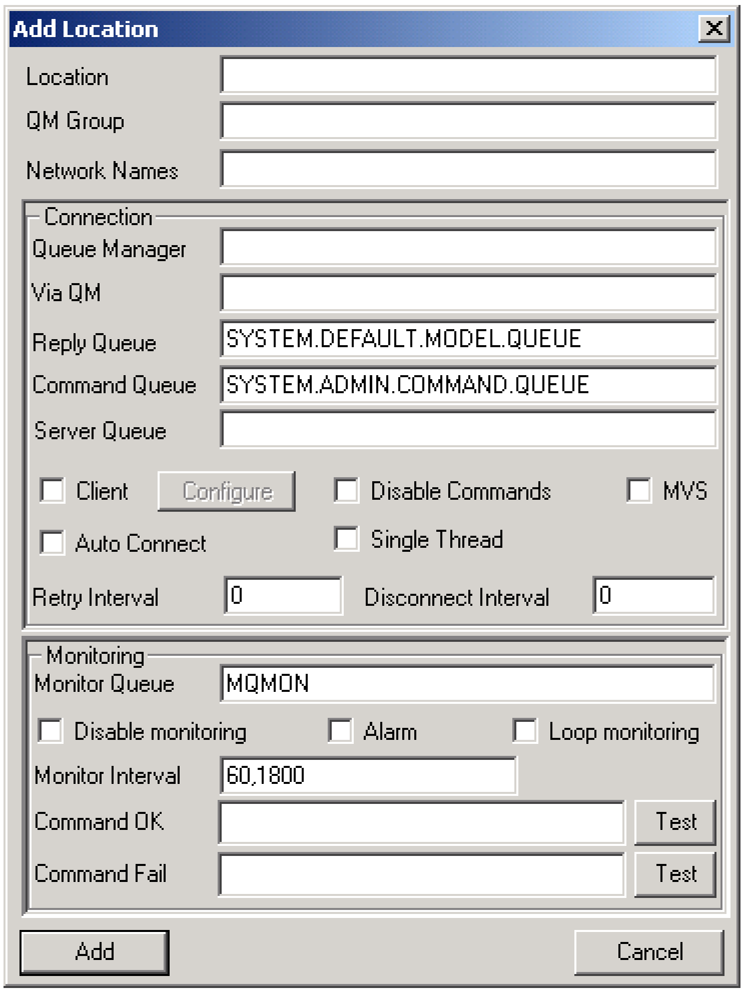
Once started, the user will be presented with a main application window with a number of menu options. If this is the first time the program has been run then the main window itself will be empty and entries for each Queue Manager must be added. By default the application receives replies from the Queue Manager via the standard model queue SYSTEM.DEFAULT.MODEL.QUEUE but it may be preferable to define an explicit queue for MQMON to use. See *"Reply Queue Details"* on page 46 for further details.



To successfully issue commands then the Queue Manager and its respective command server (eg. strmqcsv <QMNAME>) must be running.

Adding a Queue Manager Entry

To add an entry for a new Queue Manager in the Administrator use *'Add Location'* in the File menu. This will bring up a location dialog.



Reply Queue: SYSTEM.DEFAULT.MODEL.QUEUE
Command Queue: SYSTEM.ADMIN.COMMAND.QUEUE
Retry Interval: 0
Disconnect Interval: 0
Monitor Queue: MQMON
Monitor Interval: 60,1800

You can administer a local Queue Manager by simply entering its name in the Queue Manager field. To administer a remote Queue Manager you can either connect via a client or via another Queue Manager. The following examples will show the additional configuration needed to use either of these methods.

Connecting via a Client

To directly connect to the Queue Manager via a client, select the Client check box and click on the Configure button to bring up a dialog to define the client connection channel. In this dialog fill in at least the Channel Name and the Connection Name.

Connecting via a different Queue Manager

To connect via a different Queue Manager, you must first have set up the location for the directly connected Queue Manager using one of the methods above (either a local Queue Manager or via a Client). When adding a new Queue Manager entry you can then use that location in the via QM field. You must also remove the Reply Queue which is not needed for connection by this method. In addition to defining this location you must have WebSphere MQ communications in both directions between these Queue Managers.

For details of all the fields in the location dialog see *"Location Dialogs"* on page 41.

Chapter 3. Issuing Commands

MQMON provides a graphical interface to the most common commands one can issue to a Queue Manager. This interface has the advantage that the user does not need to remember the command syntax and generally reduces typing by keeping relevant fields on the dialog. To issue a command against one of the Queue Manager's via the dialogs the user must select the Queue Manager from the list and then select the item from the Commands menu that most closely approximates the command required[1].

The commands presented depend on the version of the Queue Manager at the remote location. For example clustering commands are only presented if the remote Queue Manager supports clustering.

The 'Commands' menu can be accessed either from the menu bar, or by pressing mouse button 2, after selecting the appropriate Queue Manager. Pop-up menus are available in most dialogs by pressing mouse button 2.

The menu contains commands such as :-

- a single object dialog for any Queue Manager object type
- a list dialog for any Queue Manager object type
- a single dialog and list dialog for Channel status
- queue statistics
- queue manipulation (See the section below on Queue Manipulation)
- cluster commands (where appropriate)

There are two styles of dialog for each of the main Queue Manager object types, for example, Queue and Queue List. The former style is designed to allow operations on a single object, on input of the object name. The list dialogs allow the display of, and operation on, multiple objects of that type.

The Command window

There are two types of command window, the list and single object display. Both window types allow the user to issue a command by pressing the command button if displayed at the bottom of the screen or by selecting the appropriate menu option from the pop-up menu that is displayed when you press mouse button 2. Note that only the common operations are presented as buttons, there are often more options available in the pop-up menu.

The command windows are split into the following sections :-

Fields

These fields display the current value of the objects attribute. The user can move the entry field to any attribute (except certain read-only attributes) and change it's value. This can be done by clicking anywhere on the field or its description with the mouse, using the up/down buttons or using the TAB

[1] An alternative way of entering commands to a Queue Manager is to use the MQSC window. This is available from the main menu and is described later. The MQSC window has the advantage that **any** command supported by the Queue Manager may be issued but it has the disadvantage that the output of the command is not formatted or post-processed and the user must know the exact syntax of the command.

keys. For fields that have a choice of values the drop down list must be displayed (PF4) and then the value selected using up/down keys.

- If the field is a name of another object, for example a transmission queue in a sender channel dialog then double clicking on the field will cause a dialog of that object to be displayed. This allows a quick method of displaying all the objects associated with a primary object.
- It is possible for an object dialog to contain more than one key field value. This is done by selecting more than one entry of a list dialog and then opening the definition or by ending the key field name with a '+' sign to indicate that you want to type further key values. When an object does have more than one key field specified then any command issued will be applied to all the objects in the list. Note that wherever possible the commands available will reflect the objects selected or displayed but it may be that the command selected is not applicable to all the objects. In this case an error message will be displayed. Because a single command could now generate many responses, one for each object, the response window at the bottom of the dialog is a drop down list which allows all the responses to be checked.

  For example to start (or stop) many channels at once. First display a list of channels. Select the required channels using standard controls e.g. <CTRL> + Mouse Click for individual selections, <SHIFT> + Mouse Click for selecting a range. Click Mouse Button 2 then select the command required. The command will be issued against all the selected objects in sequence and the command response is given in the drop down list at the bottom of the dialog.
- For list windows the fields are used to limit the amount of data requested from the command server. For example, on the Queue List specifying a Queue Name of "SYS*" will only request queues starting with the string "SYS".
  If space is limited the fields can be hidden by selecting the *'Hide Fields'* pop-up menu option.

Status window

The status window is used to relay information about the current status of the dialog, including any error messages. This is a drop down list so a history of responses can be seen. Note that the history will be abbreviated. Not all messages are written to the history and the same entry will not be entered twice in a row.

Action buttons

Some of the commonly used options are displayed as action buttons to the user. These are equivalent to the options on the pop-up menu.
If space is limited the buttons can be hidden by selecting the *'Hide Buttons'* pop-up menu option.

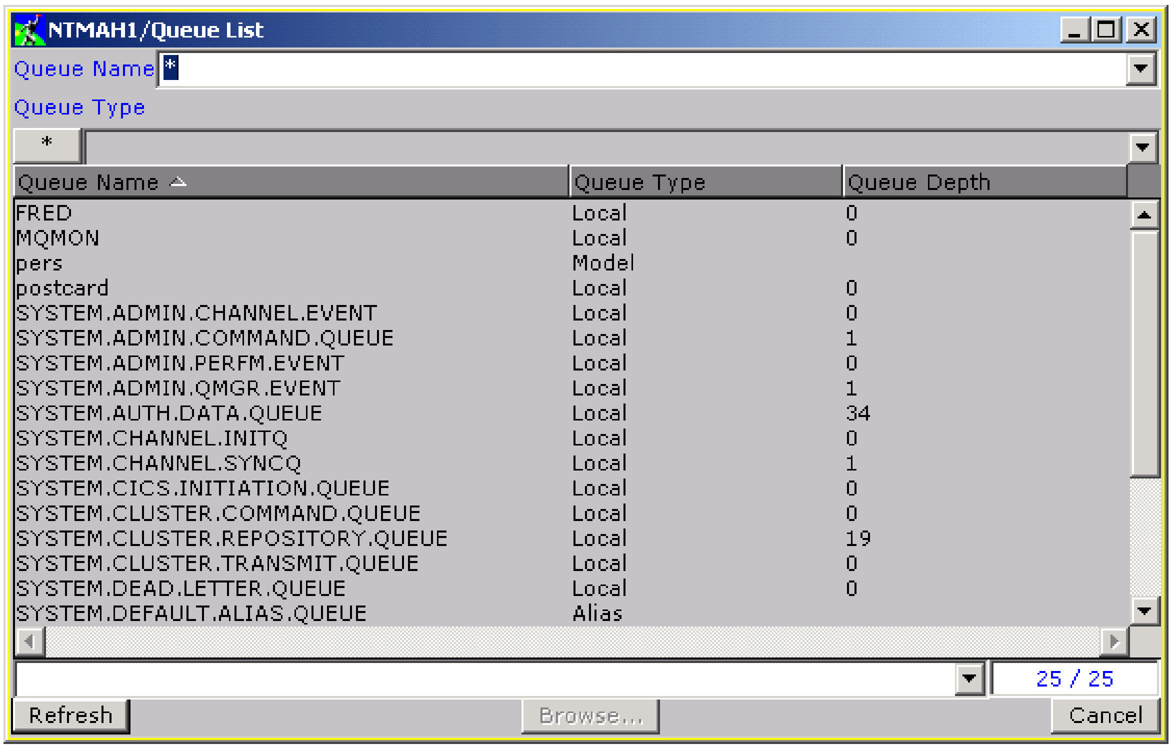
List Windows

- Object List
  A list box containing the returned objects. Double clicking on an entry will cause a dialog for that object to be shown.
- Filter button and filter entry field
  This entry field allows the user to perform filtering on the data returned by the command server before displaying. A filter can be added by typing a filter expression in the filter window and pressing the '*' filter button. The filters are stored in the list portion of the filter field, this allows

previous filter values to be selected from the drop down list. Please see *"Chapter 4. Filtering"* on page 13 for a description of the filter expressions.
If space is limited the filter can be hidden by selecting the *'Hide Filter"* pop-up menu option.

- List Titles
  When a list is refreshed the titles of the various fields in the list are displayed in the list titles window. The list of attributes displayed can be chosen from the *'Alter List'* pop-up menu option. If a particular attribute is not returned for any object in the returned list then that column is not displayed, thereby allowing more space to be used for other fields. Clicking on the list titles window will sort the list in alternatively ascending and descending order. An arrow pointing either upwards or downwards in a column title shows which field is being used to sort by and whether its ascending or descending respectively.

Examples



Queue Name: *

Queue Type: *

Queue Name	Queue Type	Queue Depth
FRED	Local	0
MQMON	Local	0
pers	Model	
postcard	Local	0
SYSTEM.ADMIN.CHANNEL.EVENT	Local	0
SYSTEM.ADMIN.COMMAND.QUEUE	Local	1
SYSTEM.ADMIN.PERFM.EVENT	Local	0
SYSTEM.ADMIN.QMGR.EVENT	Local	1
SYSTEM.AUTH.DATA.QUEUE	Local	34
SYSTEM.CHANNEL.INITQ	Local	0
SYSTEM.CHANNEL.SYNCQ	Local	1
SYSTEM.CICS.INITIATION.QUEUE	Local	0
SYSTEM.CLUSTER.COMMAND.QUEUE	Local	0
SYSTEM.CLUSTER.REPOSITORY.QUEUE	Local	19
SYSTEM.CLUSTER.TRANSMIT.QUEUE	Local	0
SYSTEM.DEAD.LETTER.QUEUE	Local	0
SYSTEM.DEFAULT.ALIAS.QUEUE	Alias	

25 / 25

Display a list of Queues

- Select the appropriate Queue Manager
- Choose *'Queue List...'* from Commands menu
- Press *Refresh* to cause the list to be shown

Defining a Queue

- Select the appropriate Queue Manager
- Choose *'Queue...'* from Commands menu
- Fill in the queue attributes required, for example *Queue Name*
  - Any blank fields will inherit the default values

- Press *Create* to cause the queue to be defined
    - Pressing *Refresh* will show the complete queue definition

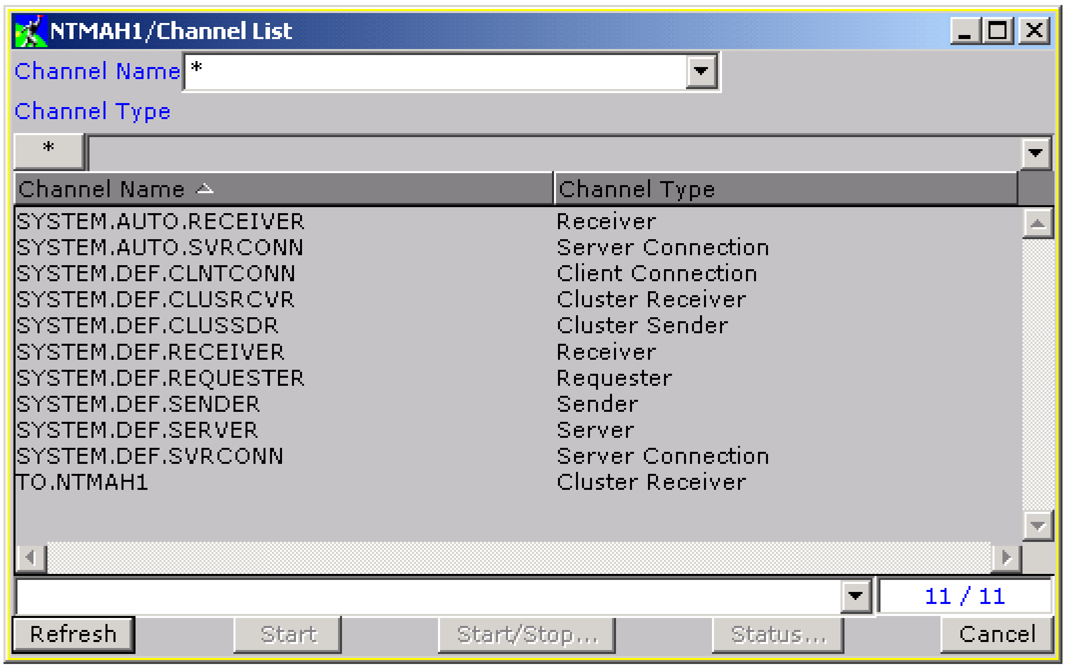
Defining a Queue like another Queue

- Select the appropriate Queue Manager
- Choose *'Queue...'* from Commands menu
- Enter queue name to copy
- Press *Refresh* to show complete queue definition
- Change any queue attributes required, which must include the *Queue Name*
- Press *Create* to cause the queue to be defined

Starting a Channel

The following examples illustrate the two different ways you could start a channel.

- Select the appropriate Queue Manager



Channel Name	Channel Type
SYSTEM.AUTO.RECEIVER	Receiver
SYSTEM.AUTO.SVRCONN	Server Connection
SYSTEM.DEF.CLNTCONN	Client Connection
SYSTEM.DEF.CLUSRCVR	Cluster Receiver
SYSTEM.DEF.CLUSSDR	Cluster Sender
SYSTEM.DEF.RECEIVER	Receiver
SYSTEM.DEF.REQUESTER	Requester
SYSTEM.DEF.SENDER	Sender
SYSTEM.DEF.SERVER	Server
SYSTEM.DEF.SVRCONN	Server Connection
TO.NTMAH1	Cluster Receiver

- Choose *'Channel List...'* from Commands menu
    - Press *Refresh* to cause the list to be shown
    - Find the Channel in question and select it
    - Select *'Start/Stop...'* from the pop-up menu or button
    - Press *'Start'* in the Start/Stop Channel dialog
- Select the appropriate Queue Manager
- Choose *'Channel...'* from Commands menu
    - Type the name of the Channel to start

- Press *Refresh* to cause the channel definition to be shown
- Select *Start/Stop...* from the pop-up menu or button
- Press *'Start'* in the Start/Stop Channel dialog

Clearly if there are a number of operations to be performed against objects of the same type then the list option involves less typing for the user.

The Pop-up menu

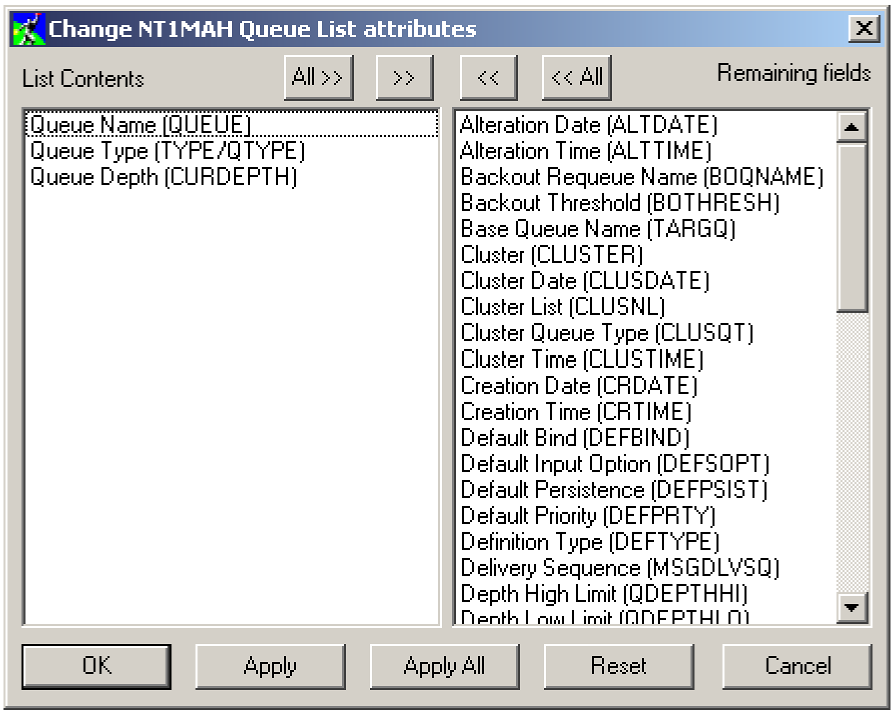
All the commands available are contained in a pop-up menu that is displayed when mouse button 2 is pressed. Care should be taken with destructive commands such as *'Delete'* and *'Clear Queue'* if confirmation dialogs are not active (see *"Preferences Dialog"* on page 38 to control confirmation dialogs). This pop-up menu may also contain a *'Print...'* option which is described in *"Printing"* on page 30, and an *'Export...'* option which is described in *"Exporting Definitions"* on page 32.

The pop-up menu also contains an *'Options'* sub menu. By selecting this the user is shown a number of options which change the look and behaviour of the dialog. Note that the list displayed will depend on the dialog. Pop-up menus for the message dialogs have other sub-menus in addition to this one. For details see *"Chapter 5. Queue Manipulation"* on page 18.

- **Alter list**
  Each list has a predetermined set of attributes that are displayed for each object. For example the Queue list displays the Queue name, Queue type and the current depth. If required, the user can change this to a list of his/her choice in two ways. Firstly the lists can be changed globally for all locations by using the option under the *'View/Set Default Lists'* menu on the main window. Secondly the list of attributes can be changed only for this location by selecting the *'Alter list'* option from the pop-up menu. See *"Alter list dialog"* on page 10 for how to operate the dialog.
- **Make Filter Default**
  Objects lists can be filtered via a Boolean expression (see *"Chapter 4. Filtering"* on page 13) to restrict the number of objects displayed in the list. Selecting this menu item will associate the current filter with this list type. Each time the dialog is initially displayed it will have this filter. Selecting this menu with the filter window empty will disassociate any previous default filter.
- **Initial Refresh**
  Selecting this menu item will cause the list to be automatically retrieved from the command server whenever a dialog of this type is opened. This should be used if the list filtering options are rarely used.
- **Auto Refresh**
  Each dialog can be set to update itself every few seconds. This allows an administrator to constantly monitor, for example, the depth of the Queues or the status of Channels. Auto Refresh is also available on the MQSC window. The window title will display the refresh rate if auto refresh is active.
- **Hide/Show Buttons**
  This option allows the user to Hide/Show the buttons of the dialog to make best use of the available screen area.

- **Hide/Show Fields**
  This option allows the user to Hide/Show the fields of the dialog to make best use of the available screen area.
- **Hide/Show Filter**
  This option allows the user to Hide/Show the filter windows of the dialog to make best use of the available screen area.

Alter list dialog



Change NT1MAH Queue List attributes

List Contents:
Queue Name (QUEUE)
Queue Type (TYPE/QTYPE)
Queue Depth (CURDEPTH)

Remaining fields:
Alteration Date (ALTDATE)
Alteration Time (ALTTIME)
Backout Requeue Name (BOQNAME)
Backout Threshold (BOTHRESH)
Base Queue Name (TARGQ)
Cluster (CLUSTER)
Cluster Date (CLUSDATE)
Cluster List (CLUSNL)
Cluster Queue Type (CLUSQT)
Cluster Time (CLUSTIME)
Creation Date (CRDATE)
Creation Time (CRTIME)
Default Bind (DEFBIND)
Default Input Option (DEFSOPT)
Default Persistence (DEFPSIST)
Default Priority (DEFPRTY)
Definition Type (DEFTYPE)
Delivery Sequence (MSGDLVSQ)
Depth High Limit (QDEPTHHI)
Depth Low Limit (QDEPTHLO)

This dialog allows the user to change the list of attributes displayed in a list window. It is invoked either by selecting the *'View/Set Default Lists'* menu on the main window or by selecting *'Alter list'* from the pop-up menu on a list window. The first of these sets the fields 'globally' for all locations. The second invocation is only 'Local' to this location and will not affect the lists displayed by other locations.

The dialog displayed shows two sets of the attributes, the **List Contents** in the left hand column and the **Remaining fields** in the right hand column. Attributes are moved either by selecting them and pressing the arrow buttons or double clicking on items. To move all the attributes use the *'All >>'* and *'<< All'* buttons.

The attributes in these lists have a text description followed by an identifier in brackets. This identifier is the name, usually the MQSC name, that should be used in filter expressions (see *"Chapter 4. Filtering"* on page 13 for more information).

The order of attributes can be set by selecting an item in the List Contents column on the left before moving the attributes across from the Remaining fields in the right hand column. Items will be added before the first selected field. If no items are selected, attributes will be added to the end of the list.

Pressing the *'OK'* or *'Apply'* buttons will cause the new attributes to be adopted by the dialog list. By default the first attribute will be used as the sort field for the list. However, if an item is selected in the List Contents column when the *'OK'* or *'Apply'* button is pressed then that field will be used as the sort field. This simple technique is overridden if the user uses SORT functions in the filter window or clicks on the list titles.

The *'Apply All'* button will update all instances of this type of list. For a global change this means that all locations will have their current list replaced by the new list in the dialog. For a local change the change will be 'saved' for this location. Subsequent invocations of MQMONNTP will display this new list for this location. This method replaces the *'Make List Default'* option available in previous releases.

MQSC Window

The MQSC window is provided to allow the user to issue commands not provided by MQMONNTP, for example, commands like **DISPLAY DQM** on z/OS. The screen is split into three main areas, the entry field at the top where you enter an MQSC command; the central area where the command server responses are displayed; and status messages, as usual, are shown at the bottom of the screen.

To make life a little easier the entry field is a drop down list box. The last 20 commands entered are kept as a history and can be recalled either by dropping down the list box or by using the up and down arrow keys. The command history will also be saved across invocations of MQMONNTP.

As with nearly all the dialogs the *'Auto Refresh'* option is available. This allows the user to type a command on the command line and have the program execute the command every few seconds. Care should be taken when doing this, however, since the response window maintains a list of **all** the responses from the command server since the start of the dialog, therefore this should be used in conjunction with the **cls** command below.

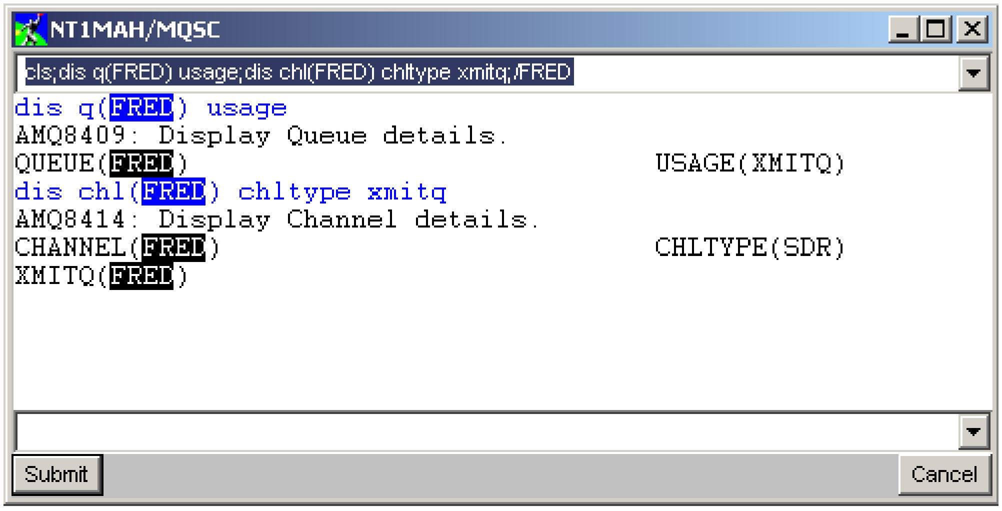
The command entered in the command field is interpreted by the program before being passed to the command server. The following additional capabilities are allowed :-

Command	**Meaning**
cls	**Clear screen command**. This command will delete all data in the main window
;	**Command separator**. Allow multiple commands to be entered on a single line Note that commands themselves should not contain this character
/	**Remove search string**. A find command with no following text will remove the highlighting of the search text.
/text	**Find text**. Issuing this command will find the next occurrence of 'text'. All instances of 'text' will be highlighted. This command stays in effect until the search string is removed by the command below
file(*filename*)	**Execute MQSC script file**. Any '?' at the end of the *filename* will bring up a file selection dialog for the path, if any, specified in *filename*.

Examples

To clear the screen, display the queue and channel called ‘FRED’ and then highlight all instances of the word ‘FRED’, use the following command :-

cls;dis q(FRED);dis chl(FRED);/FRED



```
cls;dis q(FRED) usage;dis chl(FRED) chltype xmitq;/FRED
```

```
dis q(FRED) usage
AMQ8409: Display Queue details.
QUEUE(FRED)                             USAGE(XMITQ)
dis chl(FRED) chltype xmitq
AMQ8414: Display Channel details.
CHANNEL(FRED)                           CHLTYPE(SDR)
XMITQ(FRED)
```

With auto refresh enabled the following could be useful to constantly display the state of Distributed Queuing on a z/OS Queue Manager :-

cls; dis dqm

Chapter 4. Filtering

On Queue Managers where there are many definitions of the same object type it can be difficult to 'see the wood from the trees'. What is needed is a way of limiting the objects displayed to the user, i.e. filtering. Filtering allows the user to provide a Boolean expression to determine which objects are displayed.

Note that, where possible, filtering is performed against the data cached from the command server. In other words, typing in an expression and pressing the '*' filter button will not necessarily cause a request to the command server. If all the information needed to satisfy the expression is already stored locally the filter will be performed on that information. Consequently the display of data is much faster. However, as a consequence the data displayed will not necessarily be the latest up-to-date data and users should periodically hit the *'Refresh'* button to download the latest information from the server. If the filter expression contains attributes that are not cached then a request for more information will be sent to the command server.

It is possible, in fact common, for a filter expression to contain field names that are not applicable to all entries in the list. If this is the case then the value of the field for the particular entry will be FALSE.

At the bottom right hand corner of the screen are two numbers. The first number gives the number of objects displayed in the list, and the second number gives the number of objects returned from the queue manager. If filtering is not active, these two numbers will be the same, however, with filtering there may be fewer objects in the list than were returned from the queue manager.

The expression rules are as follows :-

- **The expression is free-format**
  e.g. "2+4" and "3 + 4" are both acceptable
- **Numbers**
  Numbers can be entered as:
  - Integer 12,1234
  - Real 1.3,12345.5
  - Hex 0xFAB,0x10
- **Strings**
  Strings can be any sequence of characters between ' or " characters. The " (double quote) character can be embedded in a string by preceding it with the '\' (backslash) character. For example :-
  - "SYS.*"
  - 'ABC'
  - "This string has a \" in it"
- **Constants**
  - Colours (used in the fg() and bg() functions)
    white, black, blue, red, pink, green, cyan, yellow, brown, gray,

dblue, dred, dpink, dgreen, dgray
The constants starting with 'd' are darker versions of the colour.

- **Variables**
  Any attribute name of the object being displayed may be used in the expression. The attribute name is normally the MQSC name of the attribute and only enough of the name to ensure uniqueness need be specified. To see the identifier that should be used look in the *'Alter List'* dialog where the complete list of attributes and their identifiers are given. Note that it is not necessary to only have fields from the currently displayed list. Note also that the identifier name is not always the MQSC name since some fields don't have one. For example, Trigger Control doesn't have an actual MQSC attribute name it only has a value TRIGGER or NOTRIGGER. In this case I have introduced a value of TRIGCTL to identify this field. The same is true of fields like Harden Get Backout and Shareability.

  It is often useful to be able to temporarily see a value in the list without going through the process of actually changing the list items using *'Alter List'* described earlier. In this case you can follow the variable name with a '#' (hash) character. This will cause that field to be displayed if it's not already in the list. The columns will be displayed after the defined columns for the list in the order they were written in the expression. As soon as a filter is entered without this field followed by a hash then the field will be removed from the display

  For example, for a Queue List the following are valid variables :-

queue	Queue Name
qtype	Queue type
Usage	Usage value
Usage#	Usage Value and display Usage in the list display
curdepth	Current Queue depth
cur	Current Queue depth
Cu	Current Queue depth

- **Operators**
  Normal operator operation and precedence apply :-

+, -, *, /	Addition, Subtraction, Multiplication, Division
=,>,>=,<,<=,<>,!=	Boolean comparisons
&,|,!	Boolean AND, OR and NOT
==	String wild card matching

  The second operand is specified as a string containing the following wild card characters :-

'*'	Matches any number of characters (including 0)
'?'	Matches exactly 1 character.

- **Coercion**
  If two different operand types are involved in a sub expression the type of one or more of the operands will be changed. Most notably if strings are used in expressions other than comparisons then the string length is what is used. For example, the filter **"desc"** is TRUE only if the object has a non-blank description. So for example the filters **queue == "???"** and **queue = 3** will both display the list of queues with only three character names. As another example, **SORT(queue)** will sort the list in alphabetic queue name order but **SORT(queue+1)**, since we're forcing the queue to be treated as a number, will display the list in order of the length of their name. A similar effect is achieved by using **SORT(+queue)**.

- **Functions**
  - **all(Exp1,Exp2,Exp3,........)**
    Takes 1 or more parameters. The **all** function is used for temporarily adding a new column to a list display. For example a filter of **‘all(maxdepth#)’** will add max depth to the list of columns displayed but will still display all queue definitions regardless of whether maxdepth is applicable to that type of queue.
  - **any(Exp1,Exp2,Exp3,........)**
    Takes 1 or more parameters. The **any** function is useful for displaying fields which can contain any value. For example a filter of **‘trigctl#’** will display a column for Trigger Control but will only display those items for which the value of Trigger Control is non-zero, it is equivalent to ‘trigctl# <> 0’. There are times when you want to see the value displayed regardless of what it is. A filter of **‘any(trigctl#)’** will display those queue definitions for which Trigger Control is a valid attribute regardless of what that value is.
  - **min(Exp1, Exp2)**
    Returns minimum value of the two expressions
  - **max(Exp1, Exp2)**
    Returns maximum value of the two expressions
  - **fg(Exp, Colour)**
    Sets foreground colour, always returns TRUE
    If 'Exp' evaluates to TRUE then it sets the foreground colour of the object in the list.
    The colour can be specified using the colour constant or can be an expression.
  - **bg(Exp, Colour)**
    Sets background colour, always returns TRUE
    If 'Exp' evaluates to TRUE then it sets the background colour of the object in the list
    The colour can be specified using the colour constant or can be an expression.
  - **sort(Exp)**
    Sorts the list in ascending order of the expression. Note that since this is an explicit sort it will effectively disable sorting by the user by clicking on the list titles.
  - **sortd(Exp)**
    Sorts the list in descending order of the expression. Note that since this is an explicit sort it will effectively disable sorting by the user by clicking on the list titles.
  - **beep(Exp)**
    Causes the application to beep, always returns TRUE. If 'Exp' evaluates to TRUE then the application will issue a BEEP.
  - **alarm(Exp)**
    Causes the application to alarm, always returns TRUE. If 'Exp' evaluates to TRUE then the application will issue an alarm BEEP every few seconds provided alarms are switched on for the application.
  - **sound(Exp,Wave file)**
    Causes the application to play wave file, always returns TRUE. If 'Exp' evaluates to TRUE then the application will play the wave file.
  - **system(Exp, Command String)**
    If the expression is TRUE then the Command is executed, always returns TRUE. This is

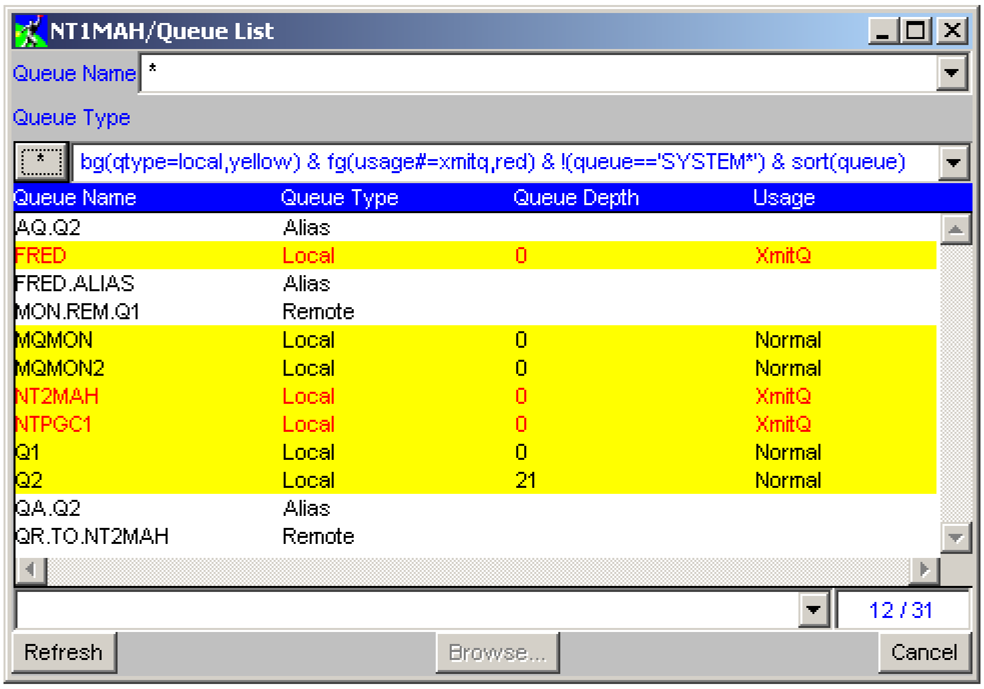
useful if you want to check for out of line situations for example a Channel being in STOPPED state or a Queue Depth being larger than 100,000. The command is the name of a .EXE program on the local machine and can be followed by parameters. The program will be started as a background process.
e.g. command(cur>100000,"LOG.EXE -t \"Queue %o is in trouble!\" ")
For more information refer to *"Command Strings"* on page 33.

Examples

On the basis that the best way to see what effect filters have is the try them, here are a number of examples (some of use and others purely for demonstration purposes). Note that the following filters are examples of filtering that can be done on the Queue List. Similar filters can be made on any of the other lists.

- **queue = "Q1"**
  Show only queue "Q1"
- **queue == "Q1*"**
  Show only queues with a name beginning "Q1"
- **queue == "*Q1"**
  Show only queues with a name ending "Q1"
- **queue == "*Q1*"**
  Show only queues with "Q1" in their name
- **queue == "????"**
  Show only queues with 4 character names
- **queue = 4**
  Show only queues with 4 character names
- **queue < 8**
  Show only queues with less than 8 character names
- **cur**
  Show only queues which have messages on them
- **cur > 100**
  Show only queues which have more than 100 messages on them
- **ipprocs# | opprocs#**
  Show only queues which are currently in-use.
  The hash characters force the relevant columns to be displayed
- **bg(usage#=xmitq,red)**
  Show all queues but highlight the transmission queues
  The hash will force the usage column to be displayed
- **fg(qtype=local,red) & fg(qtype=remote,blue)**
  Show all queues but highlight local and remote queue types
- **(qtype=local) | (qtype=remote)**
  Show only local and remote queues
- **!initq**
  Show queues which do not have an initiation queue defined.

- **trigctl#**
  Show local queues that have trigger enabled
  The hash will force the trigger control column to be displayed
- **!trigctl#**
  Show local queues that have trigger disabled
  The hash will force the trigger control column to be displayed
- **!trigctl# | !initq# | (!process# & usage# != xmitq) | !trigtype#**
  Show all the queues which have triggering disabled for whatever reason, and display any of the fields provided they have a value.
- **sortd(queue)**
  Sort the entries based on the Queue name in descending order
- **sortd(+queue)**
  Sort the entries based on the length of the Queue name in descending order
- **alarm(cur>100)**
  Switch on alarm if any queue has a depth greater than 100
- **beep(!desc)**
  Beep if there's a queue without a description
- **system(curdepth>maxdepth*0.8,"q -oQ1 -M\"%o is getting full\"")**
  If the queue is more than 80% full then issue the given command. The command given above is actually a call to my Q program (WebSphere MQ SupportPac MA01) which sends a text message to a queue. This demonstrates, in a simple way, how you can monitor the attributes of your objects and have commands issued should certain conditions arise.
- **bg(qtype=local,yellow) & fg(usage#=xmitq,red) & !(queue=='SYSTEM*') & sort(queue)**
  Combine any of the filter options together to produce the list you need.



bg(qtype=local,yellow) & fg(usage#=xmitq,red) & !(queue=='SYSTEM*') & sort(queue)

Queue Name	Queue Type	Queue Depth	Usage
AQ.Q2	Alias		
FRED	Local	0	XmitQ
FRED.ALIAS	Alias		
MON.REM.Q1	Remote		
MQMON	Local	0	Normal
MQMON2	Local	0	Normal
NT2MAH	Local	0	XmitQ
NTPGC1	Local	0	XmitQ
Q1	Local	0	Normal
Q2	Local	21	Normal
QA.Q2	Alias		
QR.TO.NT2MAH	Remote		

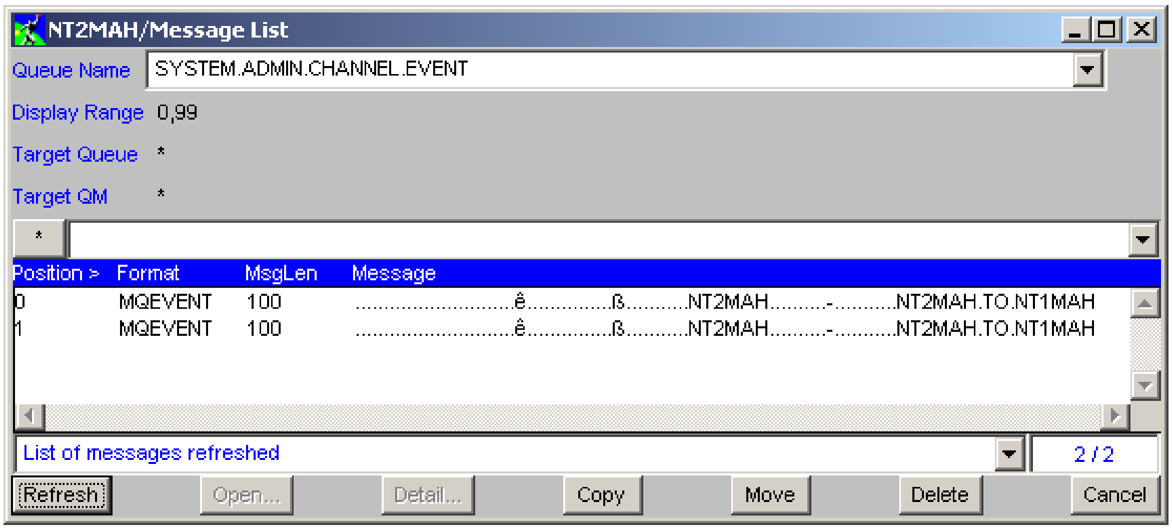
12 / 31

Chapter 5. Queue Manipulation

To manipulate messages MQMON needs direct access to the queue since no support is provided by the command server. It is possible to browse messages on other queue managers but you need a program on the remote system that understands the PCF messages generated by MQMONNTP. MQMONNTP itself clearly does understand the flows so it is quite possible to have, for example, two NT machines each running MQMONNTP browsing each others queues[2].

There are three ways of displaying the contents of a Queue.

1. **Browse Queue**
   This dialog will show the list of messages on the specified Queue



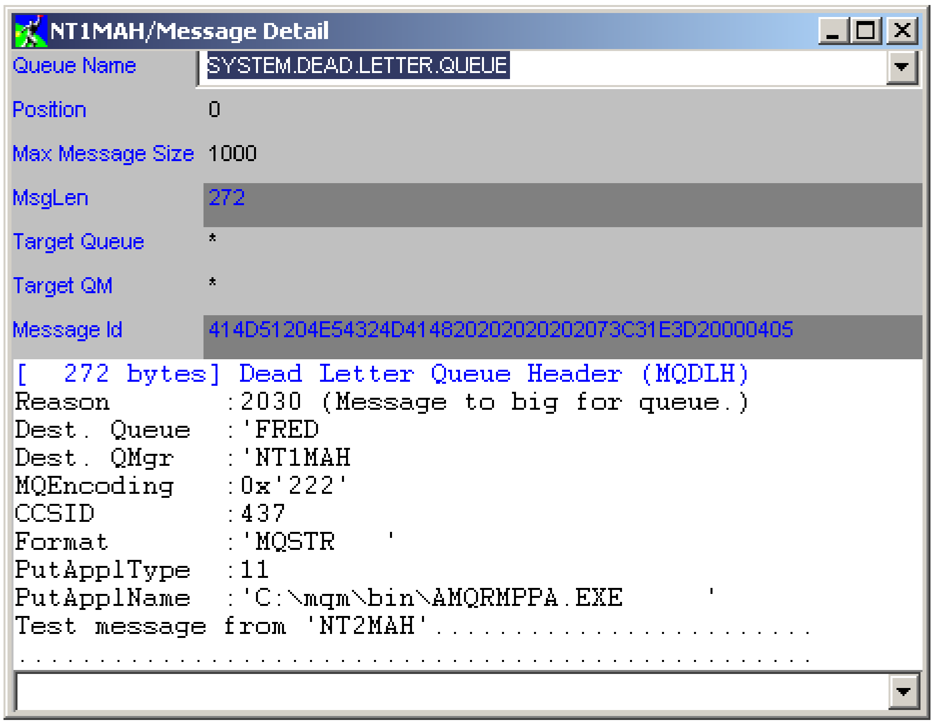
Queue Name SYSTEM.ADMIN.CHANNEL.EVENT
Display Range 0,99
Target Queue *
Target QM *

```
Position >  Format     MsgLen   Message
0           MQEVENT    100      ..........................ê...............ß...........NT2MAH..........-...........NT2MAH.TO.NT1MAH
1           MQEVENT    100      ..........................ê...............ß...........NT2MAH..........-...........NT2MAH.TO.NT1MAH
```

List of messages refreshed 2 / 2

[2] I do not provide a description of the messages but they are fairly easy to work out. It should not be too difficult to write an application on your target system which reads the queue and generates the correct responses. For simplicity, however, I suggest that most people stick to browsing queues only on the Queue Manager the application is connected to directly or via a client connection.

2. **Browse Message**
   This dialog will show the contents of the message formatted into a human readable form.

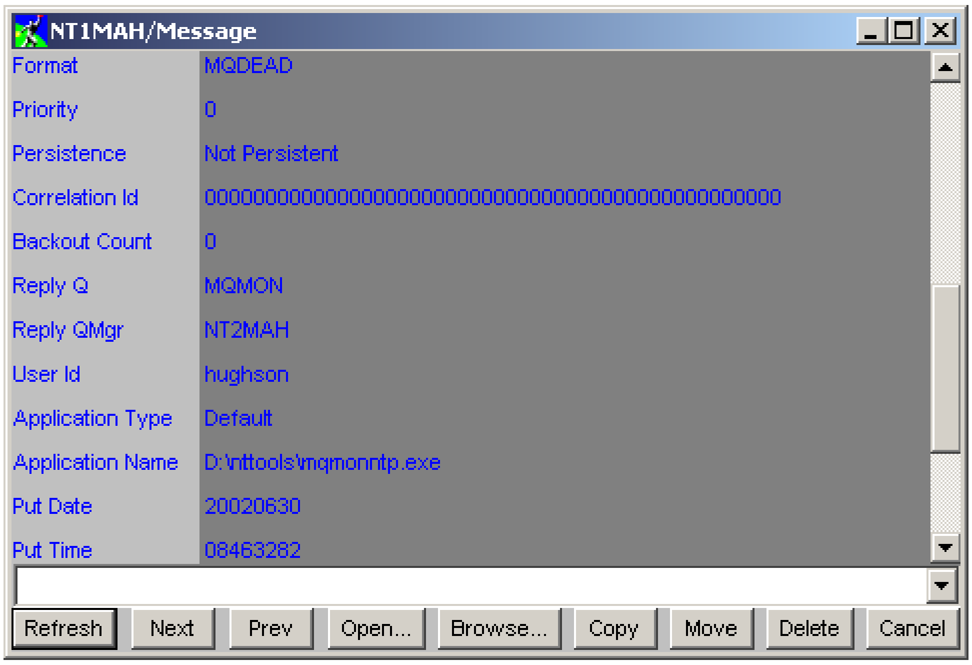


Queue Name	SYSTEM.DEAD.LETTER.QUEUE
Position	0
Max Message Size	1000
MsgLen	272
Target Queue	*
Target QM	*
Message Id	414D51204E54324D414820202020202020073C31E3D20000405

```
[  272 bytes] Dead Letter Queue Header (MQDLH)
Reason        :2030 (Message to big for queue.)
Dest. Queue   :'FRED
Dest. QMgr    :'NT1MAH
MQEncoding    :0x'222'
CCSID         :437
Format        :'MQSTR    '
PutApplType   :11
PutApplName   :'C:\mqm\bin\AMQRMPPA.EXE       '
Test message from 'NT2MAH'......................
...................................................
```

The application only understands standard WebSphere MQ formats. But it allows the user to look at messages on Dead Letter Queues, Transmission Queues and Command Queues for example. This is the most complex dialog for queue browsing. There are a number of options on the Pop-up menu which let you control what data is displayed.

3. **Message Detail**
   This dialog will show the message descriptor fields and the first few bytes of the message for a particular message on a queue



Format	MQDEAD
Priority	0
Persistence	Not Persistent
Correlation Id	000000000000000000000000000000000000000000000000
Backout Count	0
Reply Q	MQMON
Reply QMgr	NT2MAH
User Id	hughson
Application Type	Default
Application Name	D:\nttools\mqmonntp.exe
Put Date	20020630
Put Time	08463282

Identification of a message is controlled by either or both of its position and its Message Id. The position is purely an index of the message when the queue was browsed last. If messages are being added and removed from the queue while you're browsing the queue then the same message may well be given a different position number the next time you refresh the display. When performing an operation against one or more messages you can use the 'Message Selection' to determine how the application matches the selected messages against the messages on the queue. The problem is that WebSphere MQ does not have to assign a unique Message Id to each message it is under application control. Therefore when selecting a message to move the Message Id may not be sufficient, checking the position on the queue **and** the Message Id is much safer. Of course, even this is still not guaranteed to identify the same message.

Fields

The fields at the top of the dialogs allow control over the messages displayed and also controls when targeting a message to another queue.

Display Range

When a list of messages is requested you have the option to specify the range of messages you're interested in. By default the range is '0,99', which means the first 100 messages. This prevents accidentally displaying thousands of messages if the queue contains more messages than you expected !

Max Message Size

When displaying message detail only the first 1000 bytes are shown by default. This prevents very large messages causing network delays etc. However, if necessary this value can be increased to whatever size is required. If you frequently browse messages larger than 1000 bytes and find yourself constantly increasing this value then it might be worth increasing the default maximum browse message size in the preferences dialog. See *"Preferences Dialog"* on page 38 for more details.

If the message being browsed is larger than this maximum message size then clearly the message displayed will be truncated. This may cause error messages to be displayed at the end of the message reporting that the data is incomplete. A common type of message to have this problem is XML since XML messages are often fairly large and any truncation will truncate the end-tags making the message incomplete.

Target Queue Manager

When copying or moving messages to another queue you can optionally identify the target queue manager for the message, by default it has the value '*' which has the following effect :-

a. If the message has a MQXQH or MQDLH header then the queue manager is that which is specified in the header

b. If the message doesn't have a header then it is the local queue manager

Target Queue

When copying or moving messages to another queue you can optionally identify the target queue for the message, by default it has the value '*' which has the following effect :-

a. If the message has a MQXQH or MQDLH header then the queue is that which is specified in the header

b. If the message doesn't have a header then this is an error and a message will be displayed asking for the target queue name to be identified.

Pop-up Menu

The operations that may be available are listed below. Some of the most common actions can also be found as buttons.

Next

Will skip to the next message, if any, on the queue.

Prev

Will skip to the previous message, if any, on the queue.

Display Format

Will allow the user to control the amount of data that is displayed and the format in which it is displayed.

- **Formatted Message**
  If selected the application will attempt to format the message into human readable text. Once it no longer recognises the format it will display the remainder in hex.
- **Hex Message**
  If selected the message will be printed in hex.
- **Message Descriptor**
  If selected the contents of the message descriptor are displayed before the message.
- **Display Offset**
  In a HEX display the offset of the bytes from the beginning of the data is displayed.
- **Auto Detect XML Message**
  When selected the application will examine the first few bytes of the message to see whether it contains XML tags. If it does the message will be formatted as though it were an XML message. If not selected only message with the MQRFH2 format will be treated as XML.
- **Display XML shortform**
  When selected, any XML message will be displayed in an abbreviated form where not all tags and tag characters are displayed to aid readability.

- **Display Hex as chars**
  When selected, the hex fields in the MQ headers will be displayed as character strings as well as in hex. This can be useful if the Message Id or Correlation Id, for example, are character strings.

Detail Level

Will allow the user to control how much detail is shown when structures are displayed.

- **Low**
  Only the major fields of the structures are shown
- **Medium**
  The majority of structure fields are shown
- **High**
  All structure fields are shown

Copy To Clipboard

Will copy all the message text, as displayed, to the clipboard. To copy any other window to the clipboard, the standard windows action is <Alt>+<PrtSc>. This copies the whole active window.

Copy

Will copy the selected messages to the queue identified by the Target Queue and Target Queue Manager fields.

Move

Will move the selected messages to the queue identified by the Target Queue and Target Queue Manager fields.

Delete

Will delete the selected messages from the queue.

Convert

By default browse operations will convert the message to the code page and encoding values of the workstation. The codepage used to convert to can be changed in the preferences dialog if the windows codepage is not suitable for the message being browsed. By the deselecting the *'Convert'* menu option messages will not be converted when they are browsed.

Note that message copy, move and delete operations never convert the message.

Strip Headers

Messages on a WebSphere MQ queue may have a transmission queue or dead letter queue header pre-pended to the actual application message. By default it is assumed that these headers should be stripped off as a message is moved or copied to another queue. This is useful if, for example, messages on one transmission queue are to be sent to a different queue manager. In those cases you would want the existing transmission queue header to be removed and the queue manager to generate a new header when the message is put to the new location. If the messages should be copied or moved exactly as there are then this option should be turned off.

Multi Transaction

All copy, move and delete operations will be done under one transaction, this avoids the problem that a system crash or some sort of failure will leave the operation only partially completed. All the operations will either complete or the transaction will be backed out. There is a queue manager attribute, max uncommitted messages, which dictates the maximum number of messaging operations allowable by an application in a single transaction. If you are trying to move or copy more messages than this limit then clearly it can not be done in a single transaction. In these cases you must select the *‘Multi Transaction’* option and take the risk that the operation may only partially complete.

Message Selection

Operations such as Copy, Move and Delete will apply to the current message if only a single message is displayed or the selected messages if a list of messages is displayed.

A third option is to have the operation applied to all the messages on the queue. This is achieved by selecting *‘Apply to all messages’* in the *‘Message Selection’* menu. If this menu is selected then the action buttons will change to reflect the ‘all messages’ status. Because this is a potentially hazardous operation the ‘All message’ mode is removed as soon as any action is selected.

Context

When copying or moving messages between queues the context of the message will also be copied or moved by default. However, there are other options :-

- **Set all (default)**
  The context information of the copied or moved messages will be same as the original. This does require a high level of authority on the target queue.
- **Pass**
  Will pass all the context from the source message to the target message. Pass context is only valid on a move operation since pass context is not valid after a browse.
- **Default**
  Default context is used. The identity and origin context of the new messages will be that of the MQMONNTP program itself.

Chapter 6. Network View

The network view is by far the most complicated dialog in MQMONNTP. The general idea is to try and relate the various objects in the various Queue Managers.

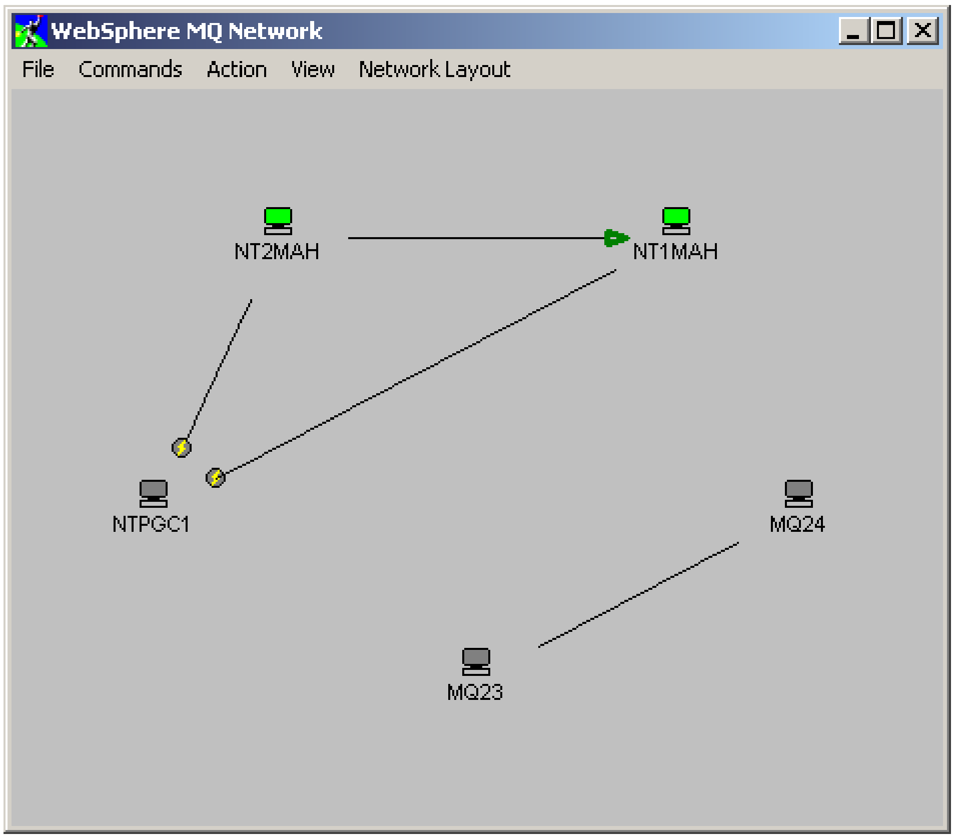
Both manually defined objects and cluster objects will be displayed in the Network View.

The main window can either display one or two windows. If two windows are displayed a sliding bar allows the user to set the proportion of the main window given to each sub-window.

A number of the windows display a hierarchical tree view known as a container. These containers are very similar to Windows container views displayed in applications such as Windows Explorer. However, an explanation of the keys available is included later in this document.

The sub-windows that can be displayed are described below.

Network Display



The Network display shows the topology of the Queue Manager network according to the channel definitions. I discovered while writing this code that displaying a set of interconnected nodes in a reliable, eye-pleasing fashion is actually very difficult. For small networks a simple circle is hard to beat. For large networks the problem becomes very difficult very quickly. Ultimately there is probably no substitute for allowing the manual placement of the Queue Managers in the display. I have therefore initially allowed three types of display selectable from the *'Network Layout'* menu.

- **Circle layout**
  This simply displays all Queue Managers in circle. All Queue Managers in the network are displayed.
- **Diagram layout**
  This layout uses the links between the Queue Managers to try an devise a reasonable looking topology. Only interconnected Queue Managers are displayed.
- **User layout**
  Allows the user to manually position each Queue Manager by clicking on the Queue Manager[3], holding the mouse button and moving it to the desired location. Note that it is much easier to do this with *'Scale'* switched off otherwise you can get some very strange results.

The other menu options available in the *'Network Layout'* are **:-**

- **Show location name/Show Queue Manager name**
  Allows control over which name is used in the display
- **Scale**
  When checked all Queue Managers will be scaled to fit on a single page. With scaling switched on moving Queue Manager at the extremes of the window can produce unexpected results.
- **Show Grid**
  When selected an array of dots is displayed. The size of the grid is determined by the Grid size specified in the Preferences dialog.
- **Snap all to Grid**
  When selected all Queue Managers will snap to their nearest grid co-ordinate. This can be useful to align Queue Managers for neatness.
- **Highlight Channels**
  When selected the channel lines from the selected Queue Manager are displayed with a thicker line in the highlight colour to make it easier to see which Queue Managers the selected Queue Manager is connected to.
- **Centre QM**
  Will move the selected Queue Manager to the centre of the network display. This can be useful if the current position of the Queue Manager is unknown.
- **Save layout**
  When you are happy with the layout for your Queue Managers you can save the current settings by selecting this option. If you want to back out subsequent changes pressing the 'User layout' option will go back to this save point. Note that the save is purely to memory. You must then press 'Save Configuration' on the main window if you want your values written to disk. As usual all changes will be written to disk if the application ends normally.
  If the layout is changed and the dialog is ended the application will automatically prompt the user as to whether the current layout should be saved.

Display Format

The network display will show the Queue Managers that have been selected as part of the 'network'. If a pair of Queue Managers have at least one channel pairing between them then a line will be drawn

[3] Note that moving a Queue Manager manually while in any layout will change the layout to User.

between the two Queue Managers showing that messages can flow between them. Only a single line will be shown regardless of how many channel pairs are defined between the Queue Managers.

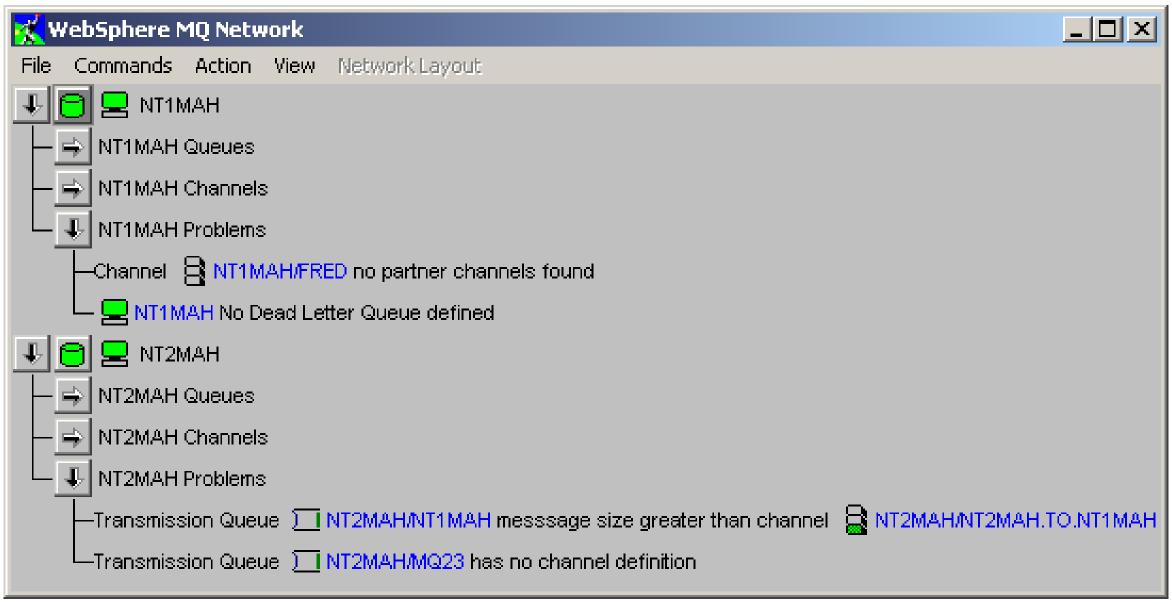
To indicate the state of the channels between the Queue Manager pair the ends of the lines may have a number of different symbols. Since a single line could represent a number of different channels it is necessary to define some order of precedence. This list is therefore given in precedence order.

• **Arrow**	At least one channel is running in direction of arrow
• **Lightening Bolt**	No running channels and at least one channel in retrying or binding
• **No entry symbol**	Channel stopped
• **No symbol displayed**	Channel inactive

See *"Appendix A: Icons"* on page 56 for examples of what these icons look like.

In the preferences dialog you can set how often the channel status information is retrieved from the Queue Managers. See *"Preferences Dialog"* on page 38 for more details.

Verify Display



NT1MAH
NT1MAH Queues
NT1MAH Channels
NT1MAH Problems
Channel NT1MAH/FRED no partner channels found
NT1MAH No Dead Letter Queue defined
NT2MAH
NT2MAH Queues
NT2MAH Channels
NT2MAH Problems
Transmission Queue NT2MAH/NT1MAH messsage size greater than channel NT2MAH/NT2MAH.TO.NT1MAH
Transmission Queue NT2MAH/MQ23 has no channel definition

This window will display the definitions that MQMONNTP has stored for each of the Queue Managers in the network[4]. Each Queue Manager contains a hierarchical display. The levels of the hierarchy are expanded by pressing on the horizontal arrow, and collapsed by pressing on the vertical arrow. The complete hierarchy is shown in the 'levels display' but essentially it contains the list of queues and channels defined for the Queue Manager.

Some of the objects will also have sub-trees to relate each object to other objects in the Queue Manager and other Queue Managers in your network. For example a remote queue will expand to show which transmission queue it resolves to, which in turn will expand to show the remote Queue Managers that the transmission queue services. Each of those Queue Managers will expand to show

[4] Note that the Queue Manager will only appear in the list when definitions are received from the Queue Manager

the channels used to get to them from this transmission queue. Once at the target Queue Manager the path resolution process continues until ultimately it will show the local queue that the remote queue resolves to.

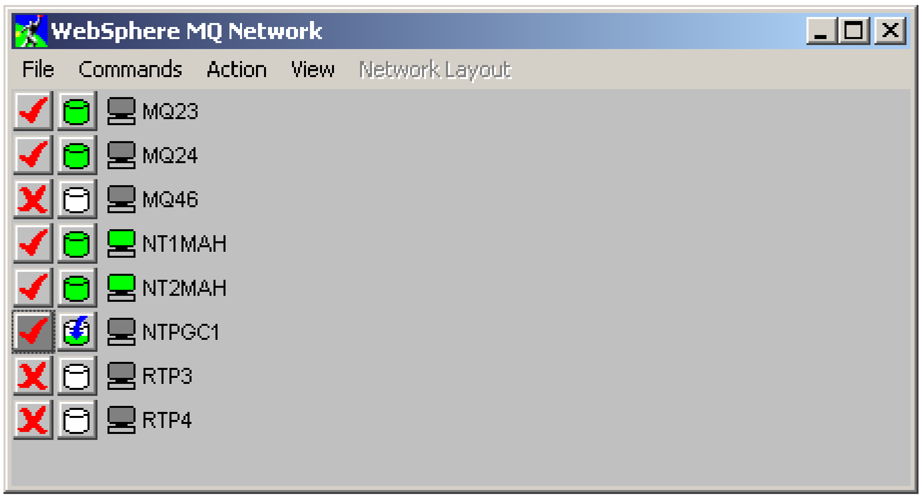
Similarly, local queues will expand if they are resolved to from another queue in the network.
Path resolution can be a complicated, time-consuming task and as such can be switched off in the Preferences dialog. For details see *"Preferences Dialog"* on page 38.

The channel objects will expand if a partner channel (i.e. one with the same name) is detected in the network.

An additional ***problems*** sub tree may be displayed after the channels sub tree if enabled in the Preferences dialog (see *"Preferences Dialog"* on page 38). The Problems sub tree contains a list of all the possible problems detected by the application in this Queue Managers configuration. Note that this is just a guide and a highlighted problem may not in fact be a problem. For example, the application expects each Queue Manager to have a defined Dead Letter Queue which is a strong MQ recommendation. There are, however, good reasons why the user may not wish to use a Dead Letter Queue. For this reason the user may use the problem selection window (see *"Problem Selection"* on page 28) to disable the checks which are not wanted. It should be pointed out that the problems displayed are not an exhaustive test, while I do intend to increase the number of consistency checks over time, there is no guarantee whatsoever that all problems will be detected.

In the object hierarchy various queue manager names, queue names and channel names will be displayed. These are really a form of push button. Clicking on the name with a mouse or pressing <space> while the object has focus will bring the up a dialog of that object allowing the user to examine its attributes and make changes if necessary.

Queue Managers Display



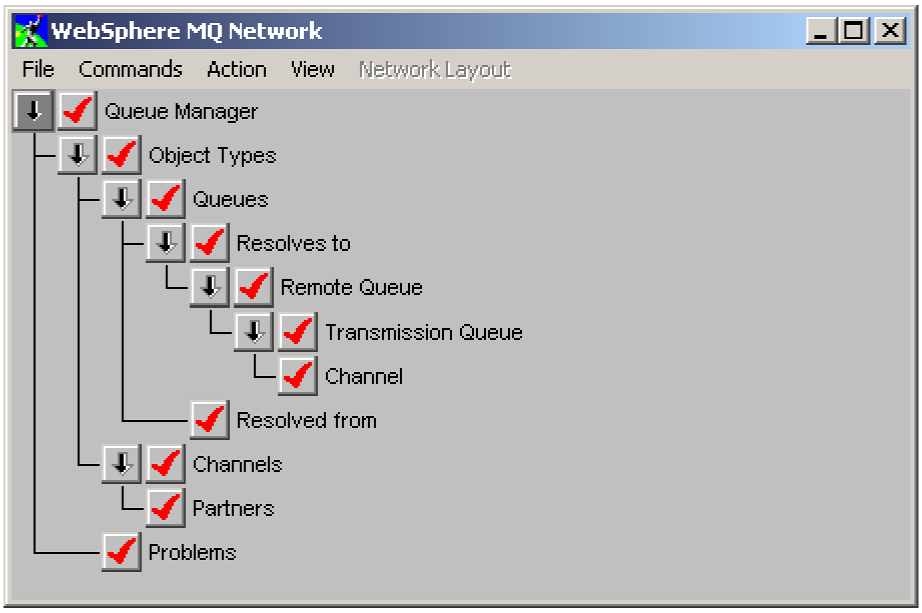
This window displays all the configured Queue Managers and allows the user to indicate which Queue Managers are part of the 'network'. The 'network' of Queue Managers is the subset of Queue Managers which will be acted upon by the Network View. Not all of your configured Queue Managers might be considered part of your immediate network and they can therefore be excluded from analysis. Each Queue Manager can either be part of or excluded from the network.

A Queue Manager is part of the network if a 'tick' appears next to its name. A Queue Manager is not part of the network if a 'cross' appears next to its name. Clicking on the icon (or pressing <space> while it has focus) will alternate between the two choices. Each location can be defined as *associated* with a number of network names. This allows many Queue Managers to be included or excluded from the network view in a single mouse click. See *"Network Names* for a description of this.

The next icon after the tick/cross indicates whether object definitions have been received from that Queue Manager. The icon can either be empty (white), full (green) or requesting (half full with arrow). At any time the user can press this button and the application will get a new set of definitions from the Queue Manager. See *"Appendix A: Icons"* on page 56 to see what these icons look like.

Levels Display

The Verify window can contain quite a complicated hierarchy. This 'Levels' display shows the various



elements of this hierarchy. By enabling or disabling various levels elements can be removed/displayed in the Verify window. Selecting and removing levels in the hierarchy only affects what is displayed, no change is made to the internal data and no data is retrieved from the remote Queue Managers.

Problem Selection

This window displays the complete list of problems the application tries to detect (see *"Appendix B: Problem Selection List"* on page 58 for the current list). By default all problems will be checked. However, by setting the icon to a 'cross' next to a problem the application will no longer check for this situation. If no problem detection at all is required a single check box in the Preferences Dialog allows all checking to be switched off (see *"Preferences Dialog"* on page 38 for details).

The list of problems is clearly an area which could be expanded in the future. I would be interested in hearing anyone's suggestions for problem checks. In particular I am interested in problems which are not easily seen from a single object definition but rely on the interaction between two or more objects.

Network Names

Each location definition has a field to specify a comma (or space) separated list of Network Names. The names themselves can be anything which makes sense for the particular Queue Manager organisation. For example, you could have each location be part of three network groups . The first identifying the type of machine, the second identifying the geographical location, the third identifying the business function. So, we could specify a value of **'iSeries, MidWest,DataCenter'**. Each name can contain any alphanumeric character but not a space since a space indicates the start of the next name.

Each unique name is displayed in the Network Names window together with a check mark. Selecting the check mark will then either include or exclude all Queue Managers with that network name from the network view. This allows easy inclusion of many Queue Managers in a single mouse click if analysis and displays only required on a subset of the Queue Managers. In the example above , for instance, it would be easy to only display the machines located in the MidWest or only the iSeries machines.

In addition to the defined network names the standard Network Name of 'All' is always available which, as its name suggests, applies to all Queue Managers.

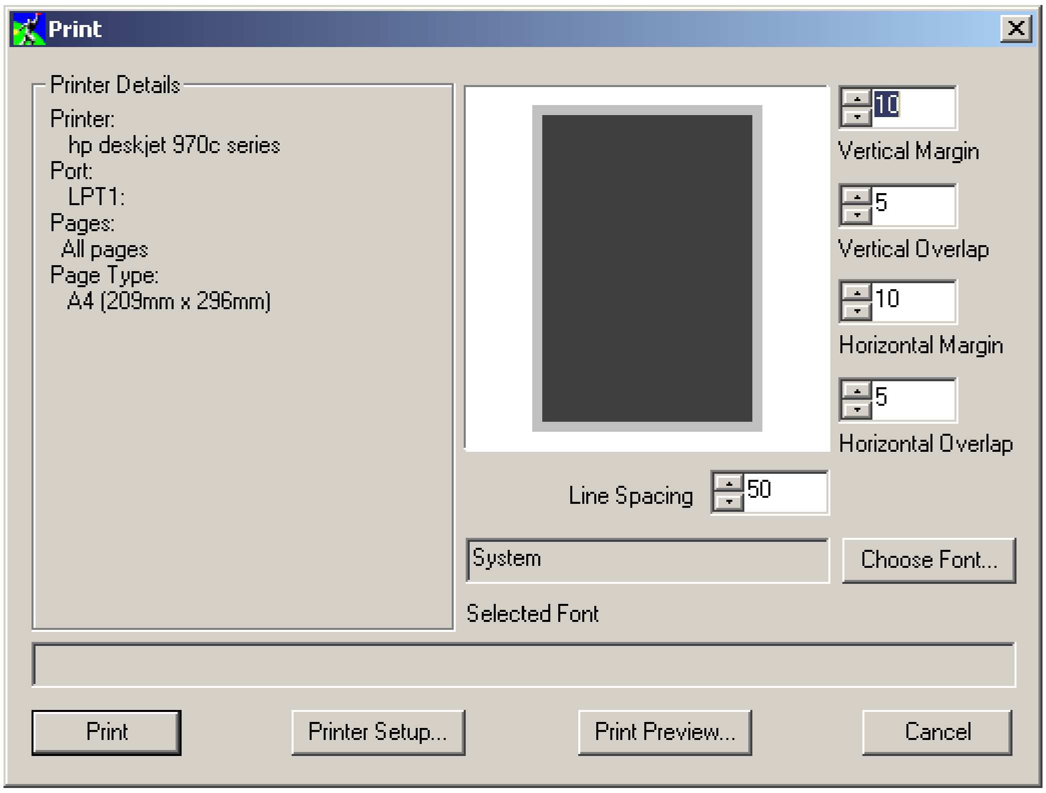
Note that Queue resolution and problem analysis applies to only those Queue Managers actually in the network view. Consequently if some Queue Managers are excluded full path and channel resolution is not possible. This may lead to problems being reported, for example, "No partner channels found", which are caused purely by the resolving Queue Manager not being included in the analysis.

Chapter 7. General Actions

Printing

The main window and the object dialogs have a *'Print...'* menu item allowing the user to send the information in the dialog to the printer. This is the first time I have ever done any printing from Windows and I have to admit it was far more involved than I'd first imagined. I have just done the bare minimum for this release. If users find it useful then I would imagine greater flexibility will be added in the future.

Selecting the *'Print...'* menu item will present a print dialog box. This dialog displays the general parameters which will be used for printing.


Print

Printer Details
Printer:
hp deskjet 970c series
Port:
LPT1:
Pages:
All pages
Page Type:
A4 (209mm x 296mm)

Vertical Margin: 10
Vertical Overlap: 5
Horizontal Margin: 10
Horizontal Overlap: 5
Line Spacing: 50
Selected Font: System

- **Printer Details**
  Displays information about the selected printer and paper size
- **Vertical & Horizontal Margin**
  Field allowing the user to set the margins on the paper. Values are in millimetres.
- **Vertical & Horizontal Overlap**
  When printing large dialogs it could be that to print all the information the output will need to be spread across multiple pages. In this case the program prints each page as though it were part of one large logical piece of paper. After printing the pieces can then be placed side by side to see then whole output.

  If multiple page output is required the Overlap values tell the program how much of an overlap is required between each page. An overlap ensures that a relatively seamless join can be made between the multiple sheets. Overlap values are specified in millimetres.

- **Line Spacing**
  This field allows the user to change the spacing between successive lines in the print output. The value is expressed as a percentage of the currently selected font height. For example a value of 50 would leave a gap of half (50%) of the height of the font between each line.
- **Choose Font**
  This button will display a dialog allowing the user to select the required font for printing. This font is not related in any way to the fonts selected for dialog screen display. The currently selected font is displayed in the box to the left of the Choose Font button.
- **Printer Setup**
  Pressing this button will bring up the system printer dialog. The exact format of this dialog is system dependent. However, it should allow the user to select the printer, the paper size, the paper orientation, pages to print and probably a whole host of printer specific options.

  On some systems the button to apply the changes in the printer dialog is ‘Print’. No printing will actually take place when this button is pressed. The printer values will be stored and returned to MQMONNTP. Only when the user presses ‘Print’ in MQMONNTP’s print dialog will any data be sent to the printer.
- **Print Preview**
  Pressing this button will display a simple window showing what the print output would look like with the current settings. This allows the user to modify various values like margins, font size, paper orientation and line spacing to ensure the data fits neatly on the page. As values are changed in the main print dialog the preview will change immediately showing the effect of the value[5]. The print preview menu allows the user to select how many pages to display and what the size of each page is relative to the size of the window.
- **Print Button**
  Only when this button is pressed will any data be sent to the printer. The button will change to a cancel print button which allows the user to cancel the print operation if necessary.

Printing Lists

When printing lists the application will print titles for the fields at the top of the upper pages. The titles shown will honour the ‘List Titles’ choice made in the list dialog window. However, there can be a slight difference with the ‘Auto’ setting. The ‘Auto’ list title setting tells the dialog to use short or full titles depending on whether there is room on the dialog. In printing there isn’t quite the same concept of ‘room on the dialog’. For printing therefore ‘Auto’ means

- **Full title** if field data length is greater than full title length
- **Short title** if full title is greater than the all the data lengths

[5] On Windows 95,98,Me the preview window is a monochrome bit map so the print colour will not be reflected in the displayed output. This is due to a bit map size limitation on these platforms.

Exporting Definitions

I was asked for an option to write the displayed dialog information to a file. There are essentially two reasons why this may be useful.

1. You want to embed the information into some other document
   In this case the information needs to be exported in human readable text format

2. You want to capture the information so the object can be defined on another Queue Manager
   In this case you really want the data exported in MQSC 'define' format.

On most of the dialog displays an export option is available in the pop-up menu. The dialog displayed allows the user to select the file name to export to and control what data is written to the file.

- **Export Type**
  As described above this option allows the user to choose between straight text format or 'MQSC DEFINE' format
- **All Objects**
  When using export from a list the default operation will export only those objects selected in the list. However, by checking the 'All Objects' box all entries in the list will be exported.
- **File Name**
  The name of the file the program will write to
- **Overwrite**
  When checked the program will overwrite any previous version of the file. Without this option checked the program will prompt the user for an overwrite decision if the file already exists.
- **Append**
  By default the program will overwrite any existing file. However, with the append box checked the data is written to end of any existing information in the file.

Export MQSC format

When data is exported only the data currently contained in the dialog is exported. This can be really useful when exporting text and only certain fields of an object should be exported. However, care should be used when exporting lists in MQSC format. Because only the data in the dialog is exported it is quite possible to export an invalid MQSC definition. To ensure a complete definition is exported select *'Alter List'* in the list window first. Ensure that all fields are in the left hand window i.e. in the **List Contents**. Select *'OK'*. This will retrieve all information about all objects. You can now export the objects from the list knowing that all data for all fields is available.

Chapter 8. Monitoring

There are two types of monitoring:-

- **Normal monitoring**
  A program on the receiving Queue Manager should send the monitor messages back to the monitor program. In other words, it should use the Reply Queue and Reply Queue Manager of the monitor message. It must set the Reply Queue and Reply Queue Manager to its own values. In addition, it must change the first character of the message from 'm' to 'r'.
- **Loop back**
  In loop back monitoring the monitor queue at the remote location must 'point' back to the Reply Queue defined in the local Queue Manager's location settings. Using this method it is not necessary to have any code other than the base WebSphere MQ product on the target system which allows very simple monitoring of any WebSphere MQ platform. Since you must define a remote queue at the remote location pointing back to the Reply Queue you can not use loop back monitoring if your reply queue is a model queue.

Using both methods a WebSphere MQ message will make a complete round trip between the monitoring program and the target Queue Manager. The application keeps track of when it needs to send a message and whether a reply is outstanding. If a message is not received within a reasonable time then the application will give a visible, and if required, an audible alert that a Queue Manager is not responding. In this way it is simple for an operator to check whether all of the Queue Managers and the Channels between them are functioning correctly.

Command Strings

A command string is a command you wish MQMONNTP to issue to the operating system. The command is always submitted as a background process. The actual format of the command and what capabilities you have depend on the OS.

The two places where you can specify a command string are :-

- **In the location dialog**
  Here you are given the opportunity to give two commands. The idea is that, during monitoring of a remote location, monitoring will either succeed or fail. As it makes the transition from one to the other the appropriate command will be issued. This allows the user to have a downstream application which is driven from MQMONNTP which takes some action when it is told that, for instance, monitoring to a remote location is now failing.
- **In the filter string using the system() function call**
  The system() function, in the filter, takes two parameters, an expression and a command string. If the expression is evaluated to TRUE the command will be issued. This allows the user to check for virtually any combination of object attributes and if they match a certain criteria then a command will be issued. Note that the system() function is executed for each item in the list returned. So, for example, if a filter of system(1,"myprog.exe") is used on a queue list then as many instances of myprog.exe will be started as there are queues in the list. You should use the expression to make sure that the program is only started as many times as necessary.

Format

The format of the command string is whatever the underlying OS supports. Essentially it should be a program name followed by zero or more parameters which are governed by the program. The Administrator will treat everything up to the first space as the program name and pass everything else as parameters. I have had varying degrees of success starting REXX and batch files. Sometimes it is necessary to start the command interpreter (CMD.EXE on NT) and pass the command you want executed as the parameter to it e.g “cmd /c rexxcmd”.
Remember that if you need embedded double quotes in a string you can precede it by a backslash.

Substitution Characters

A substitution character is a way of simply modifying the parameters passed depending on the object generating the command. Substitution will occur anywhere in the command string. The following characters are supported :-

%l : will be replaced by the location name of the location

%m: will be replaced by the Queue Manager name of the location

%o : will be replaced by the object name in the list. If the command being issued is not related to a list then it will be ignored and removed from the command string

Examples

```
"myprog.exe"

"myprog.exe -m OS2PGC1"

"myprog.exe -t "Failure for object %o on Queue Manager %m"
```

Chapter 9. Customization and Reference

Menu options

Many of the menu options act on the selected Queue Manager in the main window. The menu options available are therefore dependent on which Queue Manager is selected. Make sure that the correct Queue Manager is selected before issuing the command.

The menu options available are :-

File

- **Refresh Information**
  Causes the application to refresh its cached information about the selected Queue Manager. This should be used to update the information shown in the tree view of the main window.
- **Open Location**
  Displays a dialog showing the values set for the location allowing the current configuration to be changed. See *"Location Dialogs"* on page 41 for a description of the fields.
- **Add Location**
  Displays a dialog to allow a new location to be added to the list. See *"Location Dialogs"* on page 41 for a description of the fields.
- **Delete Location**
  Deletes the current selected location.
- **Save configuration**
  The current configuration will be written to the configuration file, MQMON.CFG.
- **Preferences**
  Displays a dialog to allow the user to change some global parameters on the way the program operates, such as saving dialog positions. See *"Preferences Dialog"* on page 38 for a description of the dialog.
- **Print**
  Print the list of Queue Manager and location names. See *"Printing"* on page 30 for more information.
- **About**
  The About dialog gives version and build date information.
- **Exit**
  Exits the program[6].
  The current configuration will automatically be written to the configuration file.

Commands

Various menu items to invoke a command dialog against the selected Queue Manager location. The list of commands available will depend on the platform and release level of the selected location.

[6] The first time the user requests to end the program it will disconnect from all the connected Queue Managers before shutting down. It is possible that this process will hang because, say, the network connection has been lost. Selecting 'Exit' again will end the program immediately bypassing the disconnect processing.

Action

- **MQSC Window**
  If commands are allowed for the selected location an MQSC window will be presented where MQSC commands can be entered and responses from the command server displayed. This has the advantage that any MQSC command supported by the command server can be entered without requiring MQMONNTP to support it directly. No parsing of the command or response is made other than formatting the data to fit the screen. See *"MQSC Window"* on page 11 for a description of how to use this window**.**
- **Enable Alarms**
  When checked the application will alarm whenever an 'alarm' situation is detected. No audible alerts will be given without this option set.
- **Reset Alarm**
  Select this option to stop the alarm sounding. The alarm will sound again if another 'alarm' situation is detected.
- **Reset Location**
  If an attempt is made to contact a location which is not available the icon for that location will show an error. This error condition is normally only reset when connection is made. However, by selecting this menu option the location can be manually reset.
- **Reset All Locations**
  As above but this menu applies to all locations in error.
- **Enable Monitoring**
  When checked the application will send messages periodically to any location which is not disabled for monitoring. See *"Chapter 8. Monitoring"* on page 33 for more details on monitoring.
- **Monitor all**
  Force the sending of monitor messages to all non-disabled locations.
- **Put Message**
  Will display a simple dialog for the selected location to allow a simple message to be put to a target queue. The message can only be a string ('MQSTR' format) but the user can choose how many messages are put and whether persistence or syncpoint is used.
- **Talk**
  Presents a dialog which allows the user to type a text message and send it to the given Queue Manager. The remote location must be running another instance of MQMONNTP or other such program which will display the message.
- **Disconnect**
  Normally the application will disconnect from a Queue Manager based on the disconnect interval specified in the location dialog. However, if the user wishes to force disconnection earlier than this then this menu may be used.

View

- **View Network**
  Displays the network view dialog. See *"Chapter 6. Network View"* on page 24 for a description of how to use this window.

- **Queue Manager name**
  Identifies the location by its Queue Manager name.
- **Location**
  Identifies the location by its location description.
- **Set Font**
  There are two fonts which can be set within the program. One used for ‘Windows’ and one for displaying ‘Data’.
  The windows font is used as the font for the main window and the command windows.
  The data font is used as the font for displaying messages on queues and in the MQSC window. It is recommended that the user select a nonproportional font, such as Courier, for the data font since displays of this type are easier to read when the columns of text are aligned on multiple rows.
- **Set Colours**
  Displays a dialog allowing the setting of the colour of various parts of the dialog windows.
- **Set Default Lists**
  Under this menu are menus for each of the list types supported by the program. Selecting a menu will present a dialog allowing the user to modify the attributes displayed when a list is selected for a location. Note that the user will be able to choose from all the attributes appropriate for **all** the locations added to MQMONNTP. However, when a list for a particular location is displayed only those attributes that are appropriate for this particular location will be shown. If an additional location is added which is a later level of Queue Manager those already defined then it may be appropriate to add new fields to the lists. See *“Alter list dialog”* on page 10 for an explanation of this dialog.
- **List view**
  Displays the list of locations in a list box.
- **Tree view**
  Displays the list of locations in a tree view. If the Preference setting “Display objects in main window” is selected then each location can be expanded to include the major objects for that Queue Manager. For channels and queues the icon changes depending on the state of the object (see *“Appendix A: Icons”* on page 56 to see what the icons look like).

  Note that the state of the object displayed will be the state the last time the information for that queue manager was refreshed. To refresh information for a location select the location and press *‘Refresh Information’* from the File menu or use the refresh icons in the network display.
- **Non-responders only**
  When this option is active only the locations which are not responding to monitor requests will be displayed. This is useful if you have many, many locations and just want to show the ones with problems.

Preferences Dialog

The preferences dialog allows the user to set the behaviour of various aspects of the application. The dialog is split into a number of different parts. The first being a set of on/off switches in a scrollable list. The item is on if a tick is displayed next to the item.

Options

- **Show Confirmations**
  When checked the application will display a confirmation dialog whenever a potentially harmful command such as “delete queue” is issued.
- **Show objects in main window**
  If checked then the locations in the main window can be expanded to show the main objects of the Queue Manager at that location. Most users use the command menu to get the lists which provide a much richer set of facilities with the returned objects.
- **Predict Hex characters**
  When browsing a queue the user can request that hexadecimal fields such as the Message Id are displayed in character form as well as in hexadecimal. It is not clear, however, whether these fields will be in ASCII or EBCDIC format. If this item is checked the application will display ASCII or EBCDIC depending on the number of characters falling into the alphanumeric range in each of the character sets. If not checked the value will be assumed to be ASCII.
- **Save object definitions**
  With this option checked the application will save the key fields in the Queue Manager objects in a file called MQMON.DFN in the program path directory. This allows the application to start with a list of the objects defined on each Queue Manager. The user must select refresh information or hit the refresh button in the network display to update this data.
- **Refresh objects when first displayed**
  When selected the application will automatically refresh the object definitions the first time the location is expanded in the main window. This requires that the ‘Show objects in main window’ is also selected.
- **Save Dialog Positions**
  When checked the positions of the command dialogs will be saved
- **Save Dialog Sizes**
  When checked the sizes of command dialogs will be saved
- **Display on taskbar**
  By default the dialog windows are displayed on the windows taskbar. However, if you have many of them then this can use up valuable taskbar space. By deselecting this option the dialogs will not be on the taskbar but can be easily accessed via the ‘Windows’ menu item on the application main window.
- **Show existing dialog by default**
  When the user requests a dialog, for example Queue List, this option controls whether any existing Queue List dialog for this location should be displayed or whether a new one should be created. Holding down the <Shift> key when the dialog menu is selected will give the alternate behaviour than specified in the preferences dialog. For example, suppose we already have a Queue List dialog for location X. With ‘Show existing dialog by default’ selected then pressing the

Queue List menu for location X will display the existing Queue List dialog. However, holding down the <Shift> key and then selecting the Queue List menu item will cause a new Queue List dialog to be created.

- **Show multi-fields in list dialog**
  Some object values can be lists such as the names in a namelist object or channel exits in a channel definition. When displaying a list of these objects the application needs to know whether to display all of the values in the list or just the first one. Normally you would want this option checked but if you have channel definitions with a large number of channel exits the list might be excessively large and so it might be worth switching this off.
- **<esc> should end dialog**
  When selected the escape key will end the current dialog.
- **Ticks should be green**

  When selected, tick symbols ✓ (sometimes known as check marks) are displayed as green. When not selected the symbol will be displayed as red. Remember than until you press 'OK' or 'Apply' no change will take place.

  Resolve Queue Paths
  In the Network Display the application will try to resolve the paths for all queues in the Queue Manager definition. For large numbers of queues and large numbers of Queue Managers this may take some time. If this information is not needed then this value may be unchecked.
- **Resolve SYSTEM Objects**
  Normally the objects starting with the string 'SYSTEM.' are only used as default definitions and are therefore not used as real objects by applications. As such you would normally not want the SYSTEM objects resolved.
- **Check for Problems**
  The Network View display can display possible problems with the current configuration of Queue Manager, for example, an alias queue that targets an undefined queue. If this information is not required then this option may be unchecked to improve performance.
- **3D List Titles**
  When checked the titles in list dialogs appear like three dimensional buttons. Regardless of this setting the titles will always behave like buttons and allow sorting based on the field title selected.
- **3D List Sort Arrow**
  For most colour schemes the arrow displayed in the list titles to identify the sorted field looks better when displayed in a 3D fashion. However, for some colour schemes it may be preferable to switch off the 3D attribute.
- **Filled List Sort Arrow**
  For the same reason as the previous option it may be preferable to colour fill the sort arrow with the foreground colour.
- **Display menu icons**
  When checked some of the menu items will have simple bitmaps drawn next to them to aid in visual recognition. Changing this option may only have an effect the next time the menu is created. It may be necessary to restart the program to see this option change reflected in all menus.

- **Alter main icon when QM failure**
  When checked the main application icon will go red if any of the Queue Managers are in error.
- **Show rule lines on lists**
  When checked list dialogs will be displayed with a ruled line underneath each item displayed in the list. The colour of this line will be the list selection background colour.
- **Use DOS font**
  There are slight differences between the Windows and DOS versions of the same codepage. The codepage itself is identified later in the preferences dialog in the locale field. This preference option just controls the characters displayed when browsing messages on queues. The right value is probably best found by trial and error.
- **Auto start command server**
  With this option checked the application will check whether the command server is running when it connects to a local Queue Manager**.** If it isn't it will be started with the 'strmqcsv' command.
- **Uppercase MODEL Queue userid**
  When using a model queue as a location reply queue the application will construct the name of the queue based on the application name and the userid running the program. For some security profiles it may be useful to always have the userid folded to uppercase.

Dialogs

- **Default Max browse message size**
  When browsing messages in a queue the message size is limited to a maximum size which is displayed on the dialog. This prevents accidentally downloading very large messages from the queue and enables quick navigation of the messages. The default maximum message size is 1000 bytes but if you frequently want to browse messages larger than this you might like to increase this default size.

Verify Network

- **Network Snap**
  The network display allows the user to move individual Queue Manager icons. The movement can 'snap' to certain locations :-

▪ **None**	no snap is performed
▪ **Qmgr**	the icon is snapped to connected Queue Managers
▪ **Grid**	the icon is snapped to a fixed size grid

- **Network Snap Size**
  The option allows the user to set the size of snap grid in pixels.
- **Auto Channel Status Update**
  The application may be configured to periodically query the running state of the channels. Which Queue Managers are queried is set by this value

▪ **Network (only connected)**	Only Network Queue Managers already connected to
▪ **Network**	All Network Queue Managers
▪ **All connected**	All Queue Managers currently connected to
▪ **All**	All Queue Managers

  - **None** Don't perform automatic channel status update

- **Auto Channel Status Update Rate**
  Sets the interval, in seconds, for channel status update

Sounds

- **Warning Sound**
  The sound played when the application wishes to issue a warning. For example an invalid value has been entered or a command failed.
  The value can be beep, none, or .WAV file.
  Depending on the OS is may be necessary to enter the full path to the .WAV file
  The application will play the sound id the user double-clicks on the entry field. This gives a quick and simple way to test the .WAV file has been specified correctly.
- **Alarm Sound**
  The sound played for the application alarm.
  The value can be as above for the warning sound.

Display

- **Locale**
  Any valid Windows locale string can be entered in this field. The value specified will control the characters displayed in dialogs such queue browsing. It will also control which characters are considered as printable. A value of 'Default' returns the application to the default windows setting.

  The drop down list contains the basic settings for this field. Selecting one and pressing apply will then display the locale selected by windows. The number at the end of the locale is the codepage. This can be changed if required provided the entire string is recognised by windows as a valid locale string.
- **Browse CCSID**
  This field allows you to specify a codepage to convert messages to when browsing messages on queues. In general if you change the value of 'locale' you would want to change this CCSID to match the codepage number. A value of 'Default' will cause the application to use the standard Windows codepage setting.

Location Dialogs

Similar dialogs are displayed when the user chooses to Add or Open a location. A description of the fields are given below. Note that the Add dialog does not contain all the fields given.

- **Location**
  A free format text description of the location.
  This field is to allow the user to assign a more meaningful name.
- **QM Group**
  A free format group name for this location
  The effect of this value is that the location will be displayed in the tree view 'under' the group name. This allows all your z/OS machines, for example, to be contained in a group such as "z/OS Queue Managers" .

Since objects are placed in the tree view in alphabetic order you can place all of your groups at the top of the main window by having a group name with a leading blank.

- **Network Names**
  You can give a comma or space separated list of Network names with which you want this location to be associated with. This allows groups of Queue Managers to be associated with particular network name. In the network view you can then include or exclude groups of Queue Managers associated with a particular name. See *Network Names*
- **Queue Manager**
  The Queue Manager name that should be used by MQMONNTP when sending messages to the remote location.
- **Via QM**
  If the administrator should not try to connect directly to the Queue Manager name given above but instead should send the messages via a different Queue Manager then this field allows the user to tell the application which Queue Manager it should use.
- **Reply Queue**
  Only required if this location is connected to directly i.e. no Via QM is specified.
  This is the name of the predefined reply queue on the above Queue Manager. It defaults to the name SYSTEM.DEFAULT.MODEL.QUEUE. See *Reply Queue Details"* on page 46 to see how another name can be used.
- **Command Queue**
  The name of the queue at the location which processes WebSphere MQ commands. This value is set automatically to the appropriate name depending on whether the MVS check box is selected or not. It can be changed to another queue if required.
- **Server Queue**
  The name of the queue at the remote machine where MQMONNTP command messages should be sent. It is not required if this location directly connects to the queue manager. But if you want to browse a queue at a remote location which you are not directly connected to, then you must have a program, for example MQMONNTP, at that location to service those requests. This is the name of the queue which that program is reading.
- **Client**
  If checked this indicates that MQMONNTP should connect to this location via a client connection rather than directly.
- **Configure**
  Only available if 'Client' is checked
  If the location is to be connected to via a client then the client definition to be used can be entered in a dialog presented when this button is pressed. If a separate client channel definition is not entered then the client connection will use the normal MQSERVER or Client Channel mechanisms to connect to the server.
  The name of this button will either be 'Configure' or 'Configured' depending on whether there is currently a client configuration specified.
- **Disable Commands**
  When checked the administrator will not send commands to that particular location. The administrator will 'beep' if the user attempts to issue a command. Monitoring to that location is still permitted. This option is useful if the remote location does not have a command server and

generating commands would result in messages being put to the Dead Letter Queue or worse causing the channel to end.

- **MVS**
  Select this if the remote location is z/OS. This will cause the name of the command queue to change. When checked, messages will be sent to the command queue in MQSC, rather than PCF format.

- **Auto Connect**
  When checked the application will always try to ensure that it is connected to the Queue Manager. If the button is unchecked then connection to the Queue Manager will be made ‘when necessary’. In other words the connection attempt will only be made when a monitor message or command is issued against this Queue Manager.

- **Single Thread**
  By default the application will communicate with a connected queue manager using two threads. One thread is used for outbound messages put to the target queues and a second thread sits in an MQGET waiting for the answers to those requests and any messages from other MQMONNTP instances. In some cases the use of a second thread may be too wasteful of system resource, for example over a client connection. In those cases you can check this box and the application will run in single thread mode. This does have some disadvantages though, most notably that messages need to be ‘expected’. It would be most inefficient if every few seconds the application issued an MQGET on the off chance of retrieving a message. Consequently it will only issue the MQGET if it has recently issued a command to the Queue Manager. Provided that you are just using MQMONNTP to issue commands to a Queue Manager and display the results you should not notice too much difference running in single thread mode. Responses may take slightly longer to be displayed since the application must now ‘poll’ for the response. If the command server does not respond within about 10 seconds or so it may be necessary to issue another command, say Refresh, to see the responses.

- **Retry Interval**
  If the connection to the remote queue manager fails for some reason (for example, the Queue Manager is ended) then this field allows the specification of a number of seconds before a reconnect attempt is made. No automatic reconnection attempt will be made if the value is 0.

- **Disconnect Interval**
  This field allows the specification of the number of seconds of inactivity before the administrator disconnects from this location. A value of 0 means that the administrator will never disconnect.

Monitoring

- **Monitor Queue**
  The name of the queue at the remote location which will receive the monitor messages.

- **Disable Monitoring**
  This option, when checked, allows the user to disable monitoring for this location.

- **Enable Alarm**
  When selected an 'alarm' event will be signalled during monitoring if this location does not respond within a given time period.

- **Loop Monitoring**
  When selected it implies that the monitor queue name is actually just a remote queue at the far location which points back to the applications reply queue.
- **Monitor Interval**
  This value determines how frequently, in seconds, monitor messages should be sent to this location. The field value can be a number of different fields, the format is as follows :-

  [OK Monitor Interval ['(' OK Expire ')'] [',' Bad Monitor Interval ['(' Bad Expire ')']]]

  The first interval gives the number of seconds between each monitor message provided responses are being received. The second interval gives the number of messages between monitor messages if responses are not being received. This can prevent a large build up of messages if a remote location is not available.

  By default the monitor messages are set to expire a few seconds after a further monitor message is to be sent out. This can be overridden, however, with an explicit value specified in the braces.

  For example the value '**30 (60),3600 (7200)**' has the following meaning :-
  If the target location is available and responding send a monitor message every 30 seconds, each with an expiry of 60 seconds. If the remote location does not respond, send a monitor message every 3600 seconds (1 hour) each with an expiry of 7200 seconds (2 hours).

  If a remote location is not available then the monitor message will probably be in a queue waiting to be delivered to that target queue manager so it is not necessary to continually send new monitor messages since the first ones are likely to be delivered eventually. Bear in mind that monitor messages are non-persistent though, so they will be lost if an intermediate queue manager is ended.

  The default value is 60,1800, note that values can not be lower than a pre-set value.
- **Last Send**
  The time the last monitor message was sent.
- **Last Receive**
  The time the last monitor message was received.
- **Average Response**
  The average response time for monitor messages.
- **Reset**
  Resets the average response time calculation.
- **Last Response**
  The number of seconds the last monitor message round trip took.
- **Command OK**
  You can enter here a command string will which be executed whenever MQMONNTP detects that monitoring to the remote location is successful. Note that the command will only be executed when there is a change of state. For example, the first time MQMONNTP receives a response from a remote location it will issue the command but each successful monitor message from there

on will not issue the command. However, should monitoring then fail the next successful monitor will cause the command to be issued.

- **Test Command OK / Test Command Fail**
  These buttons cause MQMONNTP to issue the command regardless of the state of the location. This allows testing of the command and/or syntax.
- **Command Fail**
  You can enter here a command string will which be executed whenever MQMONNTP detects that monitoring to the remote location fails. Note that the command will only be executed when there is a change of state as above.

Buttons

- **Monitor Button**
  This button can be used to force a monitor message to be sent immediately.

Container Windows

The Network Display shows a number of windows which are 'containers'. A container is a window suitable for showing text, bitmaps and buttons in a tree view. It is similar to the standard windows container control but I had a number of problems using the standard control. Firstly it was very difficult to add buttons to the control. Secondly the standard windows control is rather slow to update and paint, particularly when it has thousands of entries. For this reason I decided to write my own container control from scratch which allows me full control over the interface, features and performance.

The container can be driven either by mouse or keyboard. For keyboard control the item in focus is shown either with a slightly different background colour or with a focus rectangle around the text.

Keyboard Control

The following keys can be used to navigate the container

Arrow Keys	To move the focus from one item to another.
Space	To activate a button, selectable text or a sub tree arrow
Home	Move to top of container data
End	Move to end of container data
PageDown	Move one page down through container data
PageUp	Move one page up through container data
Shift+Home	Move to the far left of container data
Shift+End	Move to the far right of container data
Shift+PageDown	Move one page to the right in the container data
Shift+PageUp	Move one page to the left in the container data
F9	Toggle Expand/Contract state of current line
F9+Shift	Expand current line and all sub trees of current line
F9+Ctrl	Contract current line and all sub trees of current line
F9+Alt[7]+Shift	Expand all lines in container
F9+Alt+Ctrl	Contract all lines in the container

Reply Queue Details

The Administrator uses a reply queue to receive all of its reply messages from a Queue Manager it directly connects to. By default the name of this queue is SYSTEM.DEFAULT.MODEL.QUEUE[8] but can be overridden by changing the Reply Queue name in the location definition. The local or model queue of whatever name you choose must already exist on the respective Queue Manager.

If more than one instance of the Administrator is run against the same Queue Manager then each instance must use a different local reply queue or a model queue.

[7] Depending on keyboard mapping it may be necessary to use the 'Alt Gr' button rather than just the 'Alt' button.
[8] Earlier versions of the program used to set a default value of MQMON but this required that MQMON was predefined on the Queue Manager before the administrator would run against the Queue Manager.

Model Reply Queue

Using a model queue Is useful if many instances of MQMONNTP will be run against the same configuration file at the same time. The disadvantage of using a model queue as the reply is that an instance of MQMONNTP has no 'fixed address'. In other words the reply queue name is not predetermined. This in turn means that you can not use loop back monitoring since that requires a predefined remote queue.

The actual queue name used will be of the form ***MQMON.<username>.***

Colour Dialog

The colours of most of the elements of the application can be configured in the colour dialog available under the 'View' menu in the main window..

The dialog displays all the dialog elements in a list and a number of action buttons. The list of elements shows the colour of element when the dialog was started, a text description of the element, the colour of the selected scheme for this element and the current colour of the element that would be applied if the 'OK' or 'Apply' button was pressed.

The element list allows multiple selection using the standard extended selection scheme. The action buttons will either apply to those items selected or to all the elements in the list.

For example, to change to use the Grey colour scheme use the following sequence.

1. Select the Grey scheme in the colour scheme drop down list
2. Press 'Reset All to scheme' button
3. Press 'Apply' button

Extended Selection

By holding down the **'Alt'** button at the same time a selection is made from the colour list the application will select (or deselect) not only the particular item in question but all the items in the colour list which have the same current colour. This allows the user to easily change all the items that use the same colour to now use a different colour. On some systems the two **'Alt'** button behave slightly differently. One is additive to the current selections and the other will replace any previous selection.

Parameters

The Administrator program does not require any parameters but if you wish you have the option to provide a single location via command line flags. By default this location will be available for the duration of the program but will not be saved to the configuration file. You can use the **-k** flag to ask the program to store the definition in the configuration file if you wish.

The complete list of flags are :-

- **-m** < *Queue Manager Name* >
  This parameter is mandatory if you wish to create a location from the command line. All other parameters are optional and allow you to change the definition that is created.

- **-q** < *Queue Name* >
  Specify the name of the 'monitor' and 'reply' queue that should be used. If not specified the queue name **SYSTEM.DEFAULT.MODEL.QUEUE** will be assumed.

- **-l**
  Tells the program to create a client connection to this Queue Manager.

- **-a**
  Tells the program to auto-connect. In other words, the program will attempt to always have a connection to this location.

- **-s**
  Identifies the location as an MVS machine.

- **-t**
  Tells MQMONNTP to connect to the Queue Manager with only a single thread

- **-v** < *Via Queue Manager* >
  Tells the program to route all messages to this location via this Queue Manager definition. The 'Via Queue Manager' must be already defined in the programs CFG file.

- **-c** < *Channel Name* >
  For client connections you can specify explicitly the channel name to use on the connection. If this option is used then program will assume it's a TCP/IP connection and you must specify the TCP/IP Host name or address of the machine below.

- **-h** < *TCP/IP Host Name* >
  Host name of the target machine for client connections. The Channel name parameters must be specified for this parameter to take effect.

- **-g** < *Group Name* >
  Tells the program which logical group the definition belongs in

- **-k**
  By default the location created from command line options will not be saved when the program ends. However, by using this flag you can ask the program to **keep** the definition.

Chapter 10. Authorities

Defining authorisations for MQMONNTP users gives some level of control over what the application can do. For example, suppose you want some users to be able to display queues but you don't want them to go near your channels or vice versa.

Authorities are provided using a separate configuration file - MQMON.AUT. This file must be created manually (using your favourite editor) and placed in the same directory as the MQMON.CFG file. If this file is not present then all options will be made available to all users. The MQMON.AUT file can be write protected since it is constructed and updated only by the administrator setting up the authorities. Providing the application can read the file it can be protected as required. It should be noted that this mechanism is not intended to be fully secure, the serious user can always find ways to circumvent such mechanisms if required. For the majority of these kinds of situations, however, a simple way of preventing the full range of options being presented to the user is sufficient.

The best way of explaining the format of the file is to show an example :-

```
# Set authorisations for users of MQMON
#

# Global Authorisations

queue_display location_display


# Per Queue Manager Authorisations
QM: NTPGC1
      all
QM: NTPGC2
      location_change
QM: NTPGC3
      queue_change
```

The format of the file is entirely free-format. White space can be added/removed wherever the user chooses. Comments can be added from the first ';' or '#' character to the end of the line.

The file is structured into global authorisations and authorisations on a per queue manager basis. The global authorisations appear before the first 'QM:' keyword. The 'QM:' must be followed by the name of the Queue Manager and then zero or more authorisation keywords. Authorisations specified in the Queue Manager section are additive to the global authorisations. They do not replace the authorisations. So a user running with this file can :-

- Do all operations against NTPGC1
- Display queues and display and change the location for NTPGC2
- Display and change queues and display the location for NTPGC3
- Display the queue and location against all other queue managers

Note that the only queue manager that this user could display channels, processes and the queue manager object itself for is NTPGC1.

A full list of keywords for authorisations is given below.

location_create	Allows the user to create new locations
location_change	Allows the user to update this location
location_display	Allows the user to display this location
location_delete	Allows the user to delete this location
queue_create	Allows the user to create new queues
queue_change	Allows the user to change defined queues
queue_delete	Allows the user to delete queues
queue_stats	Allows the user to view queue statistics
queue_usage	Allows the user to view queue usage
queue_display	Allows the user to display queues and queue lists
queue_admin	Allows the user to issue clear queue
msg_create	Allows the user to use the put message dialog
msg_copy	Allows the user to copy messages from one queue to another
msg_move	Allows the user to move messages from one queue to another
msg_display	Allows the user to browse queues
msg_delete	Allows the user to delete messages from queues
namelist_create	Allows the user to create new namelists
namelist_change	Allows the user to change defined namelists
namelist_display	Allows the user to display namelists and namelist lists
namelist_delete	Allows the user to delete namelists
process_create	Allows the user to create new process objects
process_change	Allows the user to change process objects
process_display	Allows the user to display process objects and process lists
process_delete	Allows the user to delete process objects
channel_create	Allows the user to create new channels
channel_change	Allows the user to change existing channels
channel_display	Allows the user to display existing channels and channel lists
channel_delete	Allows the user to delete channels
channel_admin	Allows the user to issue administration commands such as reset and resolve channel.
channel_start	Allows the user to start a channel
channel_stop	Allows the user to stop a channel
channel_ping	Allows the user to ping a channel
authinfo_create	Allows the user to create new authinfo objects
authinfo_change	Allows the user to change authinfo objects
authinfo_display	Allows the user to display authinfo objects
authinfo_delete	Allows the user to delete authinfo objects

qmgr_mqsc	Allows the user to get an MQSC window. ***Care should be taken with this option since it essentially gives the user access to the full range of MQSC commands. No parsing of the command is done by the MQMONNTP program.***
qmgr_change	Allows the user to change the Queue Manager definition
qmgr_display	Allows the user to display the Queue Manager definition
qmgr_admin	Allows the user to issue administration commands against the Queue Manager such as the cluster commands RESET, REFRESH, SUSPEND and RESUME cluster.
location_all	Allows the user to do anything to the location objects
msg_all	Allows the user to do anything to messages
queue_all	Allows the user to do anything to queue objects
channel_all	Allows the user to do anything to channel objects
authinfo_all	Allows the user to do anything to authinfo objects
qmgr_all	Allows the user to do anything to the Queue Manager definition
all_create	Allows the user to create objects of all types
all_change	Allows the user to change objects of all types
all_display	Allows the user to display objects of all types
all_delete	Allows the user to delete objects of all types
all	Allows the user to do anything to all object types.

Chapter 10. Trouble Shooting

Because of the nature of messaging the application can not guarantee that replies from remote Queue Managers will be returned. If you find that a command window continually waits for a response the most common reasons are :-

1. The Command server for the Queue Manager is not running
2. The channel to the remote Queue Manager is not running
3. The channel from the remote Queue Manager is not running
4. The user does not have security access to the Queue Manager

   The usual symptom of this is that messages will build up on the remote Queue Manager’s Dead Letter Queue. The 'Q' program (SupportPac MA01) can be used to browse the messages on the Dead Letter Queue to see the reason code.

   The most common cause of this is that the userid in the incoming message does not have a security profile on the receiving machine. For Windows NT/2K it will be the logged on userid. Therefore, the receiving Queue Manager, be it OS/400, z/OS, AIX or whatever, must have a security profile for this id. Alternatively, Channel exits can be used to change the userid as the message is being transferred.

Chapter 11. Migration from previous versions

I always try to ensure that as each version is shipped, all the features that were working on the previous versions remain intact. However, since this is a spare time activity I can’t test all functions individually. When installing a new version I always recommend you backup the previous MQMONNTP program and the configuration file MQMON.CFG. If you do find a problem then please send me a description and I’ll try to fix it and send you a replacement.

Migrating from version prior to Version 5.3.2

A number of changes may have a potential affect on the operation of the program. Wherever possible the application will try to make sensible choices but it may be necessary to modify a configuration value.

List Selection Colour

Previous versions displayed selected item in the list dialogs using the reverse colours of the list entries themselves. The new version allows this colour to be set independently. It may be necessary to alter the colour setting for list selections to see the selections clearly.

Client Configuration

Previous versions always used a standard version 4 MQCD structure to make MQCONNX connections to the Queue Manager. In addition, chains of send/receive channel exits and SSL were not supported. The application will now dynamically choose the structure version . If you have trouble connecting to a location over a client connection which worked in the previous version it would be worth ensuring that your client connection definition has defined only those fields supported by your installed WebSphere MQ client.

Autorefresh resolution changed to 1 second

It may be necessary to change your AutoRefresh settings if you have any active.

Change default reply queue

To make life easier for the first time user the default reply queue is now SYSTEM.DEFAULT.MODEL.QUEUE. This may mean that anyone adding locations by starting MQMONNTP via the parameter settings may need to pass the ‘-q’ parameter to set the value back to ‘MQMON’.

Migrating from version prior to Version 5.3

No migration issues.

Migrating from version prior to Version 5.2

Options

In previous versions there used to be an Options menu which allowed you to set options such as whether the dialog positions should be saved across restarts. These configuration options are now available in the 'Preferences' dialog.

Objects in the main window

Previous versions used to allow the expansion of objects in the main window by default so that the list of queues and channels, for example, could be seen. By default this is turned off in the 5.2 version but can be selected in the preferences dialog.

List and Tree View

The default view has been changed from 'list' to 'tree'. Most users find the tree view much more flexible and appealing.

Migration from version prior to Version 5.0

Prior to Version 5.0 the application connected only to one Queue Manager and all messaging to other Queue Managers were made via this single Queue Manager. This proved restrictive and awkward since channels must be up and running between all the Queue Managers. Version 5.0 has under gone a major internal revision. Essentially what was a two threaded application has become a multithreaded application. Two connections are made to each directly connected application from two different threads. One thread is used for putting and the other for getting.

Another significant change is the ability to configure a client channel configuration as part of the location information.

The above two changes have meant that the application no longer needs to be told at start-up the name of the Queue Manager, the Queue name and/or whether to connect directly or as a client since all of these are now specified in the location information. This has meant that the format of the MQMON.CFG file has been changed. For example there is now no longer the need to have multiple 'AP:' entries. Most of this will be transparent to the user however it will be necessary to go through the locations when the program is run for the first time and say for each location whether the Queue Manager is accessed directly, via another Queue Manager or as a client. My apologies for this but it is unavoidable since I do not have the information to hand.

Migration from a version prior to Version 4.0

Functionality

The general look and feel of Version 4.0 was virtually the same as previous versions however the internal workings of the code was changed quite extensively.

Main changes

The main differences from previous versions are :-

- Main window can be displayed as a container/tree view
  The Queue Manager objects are displayed if the location is expanded.
- Loop-back monitoring
  This means that it is no longer necessary to have user code at the monitored location.
- Positions, Sizes, Filters and display options of dialogs can be saved
- User can ask for confirmations of sensitive commands
- Configuration is saved per Queue Manager
- Connections to the Queue Manager can be made via a client
- Configuration is automatically saved on application termination.

Configuration

Version 4.0 introduced a changed configuration file format. The simple line orientated file in MQMONP.LST has been replaced by a keyword orientated file called MQMON.CFG. The old file is not converted to the new file, you must add your Queue Manager locations again in the version 4.0 program. The MQMON.CFG file is an editable file and its format is given in an Appendix at the end of this document, so if you have a large number of Queue Managers and you're feeling brave you can create the file in an editor rather than going through all the dialogs. Perhaps the simplest solution would be to run the program once, add a single Queue Manager entry, then end it. This will create a file with one Queue Manager entry. You can then replicate the lines to create the other destinations you need.

Appendix A: Icons

These are the icons used in MQMONNTP, together with brief descriptions of their meaning.

Table 1: Queue Manager Icons

🖥	Queue Manager - unknown state
🖥	Queue Manager - Available state
☒	Queue Manager - Unavailable state
⧗	Queue Manager - Waiting reply
🖥	Queue Manager Group - unknown state
🖥	Queue Manager Group - No queue managers are unavailable
☒	Queue Manager Group - One or more queue managers are unavailable

Table 2: Queue Icons

▥	Queue List
▥	Local queue - Unknown depth
▯	Local queue - less than 20% full
▥	Local queue - less than 40% full
▥	Local queue - less than 60% full
▥	Local queue - less than 80% full
▥	Local queue - over 80% full
⇥	Alias queue
▥	Model queue
⇥	Remote queue

A small ‘c’ character will be displayed on the icon if the queue is representing a cluster queue definition.

Table 3: Channel Icons

▤	Channel - unknown status
🟥	Channel - STOPPED status
🟩	Channel - RUNNING status
🟨	Channel - other status (e.g. RETRYING)

Table 4: Other Object Icon

⚙	Process

☐	Empty Namelist
☐	Namelist with one name
☰	Namelist with multiple names

Table 5: Network View Icons

🟥	Queue Manager definition list empty
🟩	Queue Manager definition list populated
🟥	Queue Manager definition list refreshing
↑	One or more channels running
⚡	One or more channels retrying (none running)
⛔	One of more channels stopped (none running or retrying)

Table 6: Container Icons

↓	Sub tree expanded
↘	Sub tree expanding/contracting
→	Sub tree contracted
✓	Option selected
✗	Option deselected

Appendix B: Problem Selection List

This appendix contains the current list of problems that can be detected using the Verify Network feature in MQMONNTP. In the descriptions shown below, '%q' and '%r' represent queue names, '%c' and '%d' represent channel names, '%m' represents a queue manager name and '%l' represents either a cluster name or a cluster namelist.

- Queue %q target queue does not exist
- Queue %q is an alias to an alias queue
- Queue %q is an alias to a model queue
- Queue %q no transmission queue
- Queue %q invalid transmission queue
- Queue %q invalid Queue Manager name
- Channel %c no partner channels found
- %m No Dead Letter Queue defined
- Channel %c message size greater than DLQ %q
- Queue %q invalid initiation queue %r
- Queue %q uses an invalid process
- Transmission Queue %q has no channel definition
- Transmission Queue %q message size greater than channel %c
- %m Default transmission queue %q circular definition
- Channel %c has invalid transmission queue
- Channel %c has queue which is not a transmission queue
- Channel %c has message size greater than partner channel %d
- Queue %q initiation queue not found
- %m Invalid Dead Letter Queue defined
- %m Default transmission queue invalid
- Queue %q could not resolve to target queue
- Only one repository for cluster %l
- No repository found for cluster %l
- CLUSSDR missing for cluster %l
- CLUSRCVR missing for cluster %l
- Queue %q has an invalid clusnl %l
- Queue %q has an invalid cluster %l
- Channel %c has an invalid clusnl %l
- Queue Manager has an invalid reposnl %l

Appendix C : Configuration file

The configuration is file is a human readable, editable file called MQMON.CFG which contains a number of keyword value pairs. It is possible to construct or modify this file by hand. However, unless there is good reason to manually change the file I suggest you make all the changes using the application itself. It is well worth making a backup copy of the configuration file if it contains many settings since it is non-recoverable if it gets damaged or lost.

Note that not all keywords are used in the Windows NT version.

The format is described in a pseudo-meta language.

{}	Zero or more repetitions of what's contained in the braces
[]	optional element
|	Alternatives
<>	Supplied value
""	Literal value

```
MQMON.CFG :=  Instance
              {GlobalDialog}
              [QObjList]
              [ProcObjList]
              [ChlObjList]
              [NLObjList]
              {QMgrBlock [ClientBlock] [MQSCBlock] {DialogBlock} }

Instance :=   "AP:"
```

[x = <pos>]	X position of main window
[y = <pos>]	Y position of main window
[cx = <pos>]	Width of main window
[cy = <pos>]	Height of main window
[vx = <pos>]	X position of verify window
[vy = <pos>]	Y position of verify window
[vcx = <pos>]	Width of verify window
[vcy = <pos>]	Height of verify window
[prnsp = <value>]	Print spacing
[pmvm = <value>]	Print vertical margin setting
[prnhm = <value>]	Print horizontal margin setting
[prnvo = <value>]	Print vertical overlap setting
[prnho = <value>]	Print horizontal overlap setting
[cntr]	Show container/tree view
[qms]	Display Queue Manager names, otherwise show location description
[name]	Show 'name' container view
[icon]	Show 'icon' container view
[mini]	Show 'mini-icons' container view
[tree]	Show main window as 'tree'
[list]	Show 'main window as ‘list'
[alarm]	Alarms are active
[monitor]	Monitoring is active

[confirms]	Show confirm dialogs
[mainobjs]	Show objects in main window
[nphex]	No hex character prediction
[nsaveobj]	No Queue Manager object saving
[nauto]	No auto starting of the command server
[nmulti]	Don't show string lists in list dialogs
[nres]	No Queue path resolution
[nprob]	Don't check Queue Manager definitions for problems
[naro]	Don't auto-refresh object list when main window expanded
[ntb]	Don't display dialog windows on taskbar
[nexist]	Don't display existing dialog window by default
[savepos]	Save command dialog positions
[savesize]	Save command dialog sizes
[sys]	Resolve SYSTEM objects
[chsrate = <value>]	Channel status refresh rate
[chstype = <value>]	Channel status refresh type
[layout = <value>]	Network view layout type
[dsplev = <value>]	Verify dialog display levels
[npfont = <value>]	Nonproportional font
[snapsz = <value>]	Verify window snap size
[pom= <value>]	Preference options mask value
[po= <value>]	Preference options
[dspwnd = <value>]	Verify dialog windows displayed
[font = <fontname>]	Set window font name
[fontsize = <fontsize>]	Set window font size
[npfont = <value>]	Nonproportional font
[npfont size = <value>]	Nonproportional font size
[pfont = <value>]	Printer font
[pfont size = <value>]	Printer font size
[loc = <value>]	Current locale setting
[ccsid = <value>]	Current character set setting
[expfile = <value>]	Last used Export file
[wsnd = <value>]	Warning sound .WAV
[asnd = <value>]	Alarm sound .WAV
{ clrn = <RGB number> }	Set area colour

GlobalDialog :=	"GDG:" <DialogName> {ListBlock}
QObjList:=	"OBQ:" {<Queue Name>}
ProcObjList :=	"OBP:" {<Process Name>}
ChlObjList:=	"OBC:" {<Channel Name>}
NLObjList:-=	"OBN:" {<Namelist Name>}
QMgrBlock :=	"QM:" <QMgrName> <Description> <Machine Type> <Command Level>

[viaqm = <value>]	Via Queue Manager for this location

	[rq = <value>]	Set reply queue name for location if not 'MQMON'
	[mq = <value>]	Set monitor queue name for location if not 'MQMON'
	[cq = <value>]	Set command queue name for location if not normal value
	[sq = <value>]	Server queue name for the location
	[mvs]	Location is an MVS system
	[alarm]	Alarms are active for this location
	[auto]	Automatically connect
	[client]	Connect as a client
	[loop]	Use loop-back monitoring
	[nc]	No commands allowed
	[network]	Location is part of the 'network'
	[single]	Connect with a single thread
	[grp = <value>]	Group name for this location
	[nnames = <value>]	Network names for this location
	[disabled]	Monitoring is disabled for this location
	[retryint = <value>]	Set retry interval, if not default
	[discint = <value>]	Set disconnect interval, if not default
	[interval = <value>]	Set monitoring interval, if not default
	[cmdok = <value>]	Set command ok value for this location
	[cmdfail = <value>]	Set command fail value for this location
ClientBlock:=	“CLN:”	
	{ PCFID = Value }	PCF ID and value pairs for client definition
MQSCBlock:=	“MQS:” {<MQSC Command>}	
DialogBlock :=	"DLG:" <DialogName>	
	[x = <pos>]	X position of this dialog type
	[y = <pos>]	Y position of this dialog type
	[cx = <pos>]	Width of this dialog type
	[cy = <pos>]	Height of this dialog type
	[refresh = <value>]	Automatic refresh rate for this dialog type
	[buttons]	Display buttons on this dialog type
	[fields]	Display fields on this dialog type
	[filter]	Display filter fields on this dialog type
	[initrefresh]	Automatically refresh this dialog type
	[dftfilter = <value>]	Default filter for this list
	[ListBlock]	
	[FilterBlock]	
ListBlock :=	"LST:" <Number Attribute Pairs> { Attribute Pairs }	
FilterBlock :=	"FLT:" { <FilterString> }	

Index

0

# (hash)
    filtering, 14
/ (Find text)
    MQSC window, 11
; (Command separator)
    MQSC window, 11

A

action buttons
    command window, 6
alarm
    filtering, 15
all
    filtering, 15
any
    filtering, 15
Auto Refresh
    menu, 11

B

background colour, 15. *See Also* bg
beep
    filtering, 15
bg
    filtering, 15
button. *See* sortd
    '*' filter, 6, 13
    << All, 10
    All >>, 10
    Apply All, 11
    Create, 8
    Refresh, 7, 8, 13
    Start, 8
    Start/Stop..., 8

C

Client
    check box, 4
    connection channel, 4
cls
    MQSC window, 11
colour, 47
    scheme, 47
Command String
    filtering, 15
command window, 5

action buttons, 6
fields, 5
Filter button, 6
filter entry field, 6
status, 6
configuration file, 35
MQMON.CFG, 2
confirmation dialogs, 9
container, 24

D

DISPLAY DQM, 11
Display Range, 20

F

fg
filtering, 15
fields
command window, 5
file
MQSC window, 11
Filter button
command window, 6
filter entry field
command window, 6
filtering. *See* sort
# (hash), 14
alarm, 15
all, 15
any, 15
beep, 15
bg, 15
Command String, 15
fg, 15
max, 15
min, 15
sort, 15
sortd, 15
sound, 15
system, 15
foreground colour, 15. *See Also* fg

I

Installation, 2
configuration file, 2
object definition file, 2

L

location dialog, 4

M

max
- filtering, 15

Max Message Size, 20
max uncommitted messages, 23
menu
- About, 35
- Add Location, 4, 35
- Alter List, 7, 9
- Apply to all messages, 23
- Auto Refresh, 9, 11
- Channel List..., 8
- Channel..., 8
- Circle layout, 25
- Clear Queue, 9
- Commands, 5
- Context, 23
- Convert, 22
- Copy, 22
- Copy To Clipboard, 22
- Delete, 9, 22
- Delete Location, 35
- Detail, 22
- Diagram layout, 25
- Disconnect, 36
- Display Format, 21
- Enable Alarms, 36
- Enable Monitoring, 36
- Exit, 35
- Hide Buttons, 6, 9
- Hide Fields, 6, 10
- Hide Filter, 7, 10
- Initial Refresh, 9
- List view, 37
- Location, 37
- Make Filter Default, 9
- Make List Default, 11
- Message Selection, 23
- Monitor all, 36
- Move, 22
- MQSC window, 36
- Multi Transaction, 23
- Network Layout, 24
- Next, 21
- Non-responders only, 37
- Open Location, 35
- Options, 9
- Preferences, 35
- Prev, 21

Print..., 30, 35
Put Message, 36
Queue List..., 7
Queue Manager name, 37
Queue..., 7, 8
Refresh Information, 35
Reset Alarm, 36
Save configuration, 35
Save layout, 25
Scale, 25
Set Colours, 37
Set Default Lists, 9, 10, 37
Set Font, 37
Show Grid, 25
Show location name, 25
Show Queue Manager name, 25
Snap all to Grid, 25
Strip Headers, 22
Talk, 36
Tree view, 37
User layout, 25
View Network, 36

min
filtering, 15

MQMON.DFN
object definition file, 2

MQSC window, 11
/ (Find text), 11
; (Command separator), 11
cls, 11
file, 11
status, 11

N

Network display, 24
Network Names, 29

O

object definition file
Installation, 2
MQMON.DFN, 2

P

Parameters, 48

R

Reply Queue, 46

S

sort
    filtering, 15
sortd. *See* sort
    filtering, 15
sound
    filtering, 15
status
    command window, 6
    MQSC window, 11
system
    filtering, 15
system(), 33
SYSTEM.DEFAULT.MODEL.QUEUE, 46

T

Target Queue, 21
Target Queue Manager, 20

W

WebSphere MQ client, 2