

MO31: MQSeries for OS/2 Remote Queue Administrator
MO71: MQSeries for Windows NT Remote Queue Administrator
User Guide
Version 4.5

7th April 1999

Paul Clarke
MQSeries Development
MP206,
IBM UK Laboratories Ltd.
Hursley
Winchester
Hants, SO21 2JN
United Kingdom

clarkep@hursley.ibm.com

Take Note!

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

Sixth Edition, April 1999

This edition applies to Version 4.5 of MO31: MQSeries for OS/2 Remote Queue Administrator and MO71: MQSeries for Windows NT Remote Queue Administrator and to all subsequent releases and modifications until otherwise indicated in new editions.

(c) Copyright International Business Machines Corporation 1999. All rights reserved.

Note to U.S. Government Users - Documentation related to restricted rights - Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corporation.

Notices

The following paragraph does not apply in any country where such provisions are inconsistent with local law.

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates.

Any reference to an IBM licensed program or other IBM product in this publication is not intended to state or imply that only IBM's program or other product may be used. Any functionally equivalent program that does not infringe any of the intellectual property rights may be used instead of the IBM product. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, New York 10594, USA.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The use of the information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item has been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

MQSeries
IBM
Operating System/2
OS/2
AIX
MVS

The following terms are trademarks of other companies in the United States and/or other countries:
Windows NT - Microsoft Corporation

Notices	Page 3
MQSeries Remote Queue Administrator	Page 6
Main changes from previous version	Page 6
Overview	Page 6
Installation	Page 8
Create a reply queue	Page 8
Running the Administrator	Page 8
Menu options	Page 9
Location Dialogs	Page 11
Issuing Commands	Page 12
Queue Browsing	Page 13
The Command window	Page 14
The Pop-up menu	Page 14
Filtering	Page 16
Examples	Page 19
Command Strings	Page 21
Format	Page 21
Substitution Characters	Page 21
Problem determination	Page 22
Fixed problems	Page 22
Known problems	Page 22
Migration from previous versions	Page 23
Migration from a Version 4.0 or later	Page 23
Migration from a version prior to Version 4.0	Page 23
Functionality	Page 23
Main changes	Page 23
Configuration	Page 23
Configuration file	Page 24

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 (Programmatical Command Format) messages and since MQMON already had the basic structure required I decided to add command support to it. The resulting program allows you to issue virtually all MQSC commands 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.

I would 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. Because of this it is likely that this might be the last release of the SupportPac. I can't really offer, as a spare time activity, anything over and above what is commonly available.

MQSeries Remote Queue Administrator

This document describes the functions available in the SupportPac.

Main changes from previous version

The main changes are :-

1. Clustering object support

When MQMON is administrating a Queue Manager which support MQSeries clusters additional commands and object fields will be available. No change will be visible on old Queue Managers.

2. Multi-object support

It's now possible to select multiple items in the object lists. Selecting a command will issue the command against all the objects. Similarly, in the object dialogs, by typing a '+' character after the object name the dialog will allow many object names to be specified.

3. List sorting

I've added the ability to sort the object lists by clicking on the list titles. The list will be sorted by the list item clicked on. Successive clicks will sort in ascending or descending order.

4. Attribute key words

On list dialogs, where you can type an expression to filter what is displayed, it was unclear what the names of the various attributes were. To help with this the 'Alter List' dialog will now display not only the field names but also the field identifiers.

5. Colour and Font support

You can now specify from the 'View' menu the Font and Colours that should be used.

The colour dialog allows you to specify the colours for the various parts of the dialogs. It is known that some values, for example, menus, do not have any affect on the application.

6. Command support

You can now add commands to be issued in the filter expression and in the location dialog. This allows for OS commands to be automatically run when certain conditions arise. For example, if monitoring to a location fails you can automatically run a program. See the section on Commands for more information.

7. Command graying

To make it more clear what commands are allowed the commands that are not currently available, given the current object selection, are grayed out.

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 MQSeries channels. The user can then perform the following functions against the Queue Managers in the list :-

1. Issuing 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 MVS). The list of commands is also available by pressing mouse button 2.

2. Browsing messages on a local queue

The queue being browsed can be either on the local Queue Manager or a remote Queue Manager provided a program like 'MQMON' is running on the remote Queue Manager. Running MQMON as a client will allow browsing of queues to any remote Queue Manager since, as far as MQMON is concerned, the Queue Manager is local.

3. 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.

4. Monitoring

With monitoring enabled, the program will periodically send 'monitor' messages to any enabled Queue Manager. There are two types of monitoring, in either case the configuration must identify a target queue on the remote location for the application to send monitor messages to.

▪ 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 replyQ and replyQM of the monitor message. It must set the replyQ and replyQM 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 administrator application. By default this will be 'MQMON' on the local Queue Manager. Using this method it is not necessary to have any other code than the base MQSeries product on the target system which allows very simple monitoring of any MQSeries platform.

- Using both methods an MQSeries 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 his Queue Managers and the Channels between them are functioning correctly.

Installation

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

- OS/2
 - MQMONP.EXE
 - MQMON.EXE
 - MQMONP.HL
- Windows NT
 - MQMONNTP.EXE

Create a reply queue

The Administrator uses a single queue to receive all of its reply messages. By default the name of this queue is MQMON but can be overridden by the -q parameter to the program. This queue must be created before the program is started.

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

Ensure the queue manager is running.

Ensure the command server is running (eg.STRMQCSV <QMNAME>)

Running the Administrator

To run the Administrator enter the following :-

- (OS/2) Type 'mqmonp'
- (Windows NT) Type 'mqmonntp'

Followed optionally by any or none of the parameters below :-

- -m QMNAME
By default the program will run against the default queue manager. If you want to run against an explicit queue manager then use this parameter.
- -q QUEUENAME
Use this parameter to specify the name of the reply queue. By default the application will assume a reply queue of MQMON.
- -lmqic or -lmqic32
By default the application will run directly against the Queue Manager libraries. If you want to run as a client to a remote queue manager then use this parameters to specify the name of the client library you wish the program to connect to. This can be any library exporting the MQI so -lmqm is a valid value but this is the default.

- -n

The Application creates a second thread which receives all the replies from the other Queue Managers. If you're running as a client you may find that two connections to the server is too expensive. This parameter tells the program not to create a second thread for the replies. Instead the application will poll the replies queue periodically for replies. The application will not be quite as responsive in this case.

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

If the program runs successfully the user will be presented with a main application window with a number of menu options and a list of one or more queue managers. If this is the first time you've ever run the program then the list will contain just an entry for the current queue manager you're connected too. The menu options available are described below.

Menu options

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.
 - Open Entry
Displays a dialog showing the values set for the location. This allows the values to be changed, see description below.
 - Add Entry
Displays a dialog to allow a new location to be added to the list.
 - Delete Entry
Deletes the current selected location.
 - Save configuration
The current configuration will be written to the configuration file.
 - Exit
Exits the program. 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. Commands are describe in the next section.
- Action
 - Show Network
This feature has been turned of in the current version

- **Enable Alarms**
When checked the application will alarm whenever an 'alarm' situation is detected. No audible alerts will be given without this option set.
- **Enable Monitoring**
When checked the application will send messages periodically to any location which is not disabled for monitoring.
- **Reset Alarm**
Select this option to stop the alarm sounding. The alarm will sound again if another 'alarm' situation is detected.
- **Monitor all**
Force the sending of monitor messages to all non-disabled locations.
- **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 the Administrator or other such program which will display the message. This option should not be used for 'loop-back' locations since any message sent to the monitor queue will just be returned.
- **View**
 - **List view**
Displays the list of locations in a list box.
 - **Container view**
Displays the list of locations and objects in a tree view.
 - **Non-responders only**
When this menu 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 filter out the ones with problems.
 - **Queue Manager name**
Identifies the location by its Queue Manager name. It may be necessary to chain the view type from List to Container view before the effect is seen.
 - **Location**
Identifies the location by its location description. It may be necessary to chain the view type from List to Container view before the effect is seen.
 - **View options (OS/2 only)**
On OS/2 the container view can be modified by selecting various display options such as icon view, tree view. Use combinations of these options to select the view you want.
 - **Choose Font**
Displays a dialog allowing the setting of the font used in the main window and the command windows.

- Choose Colour
Displays a dialog allowing the setting of the colour of various parts of the dialog windows.
- Options
 - Save Options
 - ♦ Save dialog positions
When checked the positions of the command dialogs will be saved.
 - ♦ Save dialog sizes
When checked the sizes of the command dialogs will be saved.
 - Show confirmations
When checked the application will display a confirmation dialog whenever a potentially harmful command is issued.
- Help
 - Help options (if any)
Help is not supported on Windows NT and to a very limited degree on OS/2. However, after a few minutes I hope that you will find the interface intuitive enough to not require any assistance.
 - About
The About dialog gives version and build date information.

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.

- Network
This option is currently not supported.
- Disable Commands
When checked the application will not send commands to that particular location. The application 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 MVS. 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.
- Location
A text description of the location. This field is to allow the user to assign a more meaningful name.

- **Queue Manager**
The Queue Manager name that should be used by the application when sending messages to the remote location.
- **Command Queue**
The name of the queue at the location which processes MQSeries commands. This value is set automatically when the user selects or deselects the MVS button. It can be changed to another queue if required.
- **Monitor Queue**
The name of the queue at the location which will receive the monitor messages.
- **Disabled**
This option allows the user to disable monitoring for this location.
- **Loop**
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.
- **Alarm**
When selected an 'alarm' event will be signaled during monitoring if this location does not respond within a given time period.
- **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.
- **Monitor Interval**
How frequently, in seconds, monitor message should be sent to this location. This value can not be lower than a preset value.
- **Command OK**
You can enter here a command string which will be executed whenever MQMON 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 MQMON 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**
This button causes MQMON 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 which will be executed whenever MQMON 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.
- **Test Command Fail**
< as for Test Command OK >
- **Monitor Button**
This button can be used to force a monitor message to be sent immediately.

Issuing Commands

To issue a command against one of the Queue Manager's the user must select the Queue Manager from the list and then select the menu item from the Commands list that closely approximates the command required.

The menu contains commands :-

- for a single object dialog and a list dialog for all of the three Queue Manager objects, e.g. Queues, Channels and Processes.
- for a single dialog and list dialog for Channel status.
- for examining the messages on a queue
See the section below on Queue browsing.

So, for example, suppose I wish to 'start a Channel'. I have two ways of doing it :-

1. Select 'Channel List...'

- Press REFRESH to cause the list to shown
- Find the Channel in question and select it
- Press mouse button 2 to display the context sensitive menu
- Select start

2. Select 'Channel...'

- Type the name of the Channel to start
- Press mouse button 2 to display the context sensitive menu
- Select start

Obviously if there are a number of operations to be performed against Channels then the list option involves less typing for the user.

Queue Browsing

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

2. Browse Message

This dialog will show the message descriptor fields and the first few bytes of the message for a particular message on a queue

3. Message Detail

This dialog will show the contents of the message formatted into a human readable form. Obviously the application only understands standard MQSeries 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.

Identification of a message is given by it's position. This is purely index of the message when the queue was browsed last. If messages are being put and got from the queue while you're browsing then the same message may well be given a different position number the next time you refresh the display.

The commands Next and Prev provide a convenient way of traversing the queue by incrementing and decrementing the position number respectively.

The current implementation has a number of restrictions :-

1. Only the first 100 messages on a queue can be browsed.
2. Only the first 4000 bytes of a message can be examined.

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 MQMON. MQMON itself obviously does understand the flows so it is quite possible to have, for example, two NT machines each running MQMONNTP browsing each others queues. 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.

The Command window

There are two types of command window, the list and single object display. Both dialog 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. The 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 keys. For fields that have a choice of values the combo-box must be displayed (PF4) and then the value selected using up/down keys.

- 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 combo-box which allows all the responses to be checked.
- For list dialogs 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.
- Object List (list windows only)
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 (list windows only)
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 the section on filtering for a description of the filter expressions. If space is limited the filter windows 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 all objects in a returns 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. A '>' or '<' character after the column titles shows which field is being used to sort by and whether its ascending or descending.
- Status window
This 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.
- Action buttons
Some of the commonly used options are displayed as 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.

The Pop-up menu

As you can see all the commands are contained on the 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. The popup menu also contains an Options entry. By selecting this the user is shown a number of options which change the behavior of the actual dialog. Note that the list displayed will depend on the dialog.

- 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 by selecting this menu item.

A dialog is displayed showing those attribute that are 'in' the list and those that are left to be selected. Attributes are moved either by selecting them and pressing the arrow buttons or double clicking on items. The attributes are generally a test description followed by an identifier in brackets. This identifier is the name that should be used in filter expressions (see the section on filtering for more information).

The order of attributes can be set by selecting an item in the 'in' list before moving the attributes across. 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 'in' list when the 'OK' or 'Apply' button is pressed then that field will be used as the sort field. This is a simple technique and is overridden if the user uses SORT functions in the filter window or clicks on the list titles.

Note that the user list selection will not be saved in the configuration file unless the 'Make List Default' menu is selected.

- Make List Default

Selecting this option will save the currently defined list as the default list to be displayed for this object type.

- 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.

- 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.

The following commands are available on the Message Detail dialog to allow the user to control the amount of data that is displayed.

- Message Descriptor

If selected the contents of the message descriptor are displayed before the message.

- Hex Message

If selected the message will be printed in hex.

- Formatted Message

If selected the application will attempt to format the message into human readable text. Once it no longer recognizes the format it will display the remainder in hex.

- Low Detail, Medium Detail, High Detail

These commands control which fields of the various structures are displayed.

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 type a Boolean expression which is evaluated for every object returned from the command server. Only those objects which evaluate to TRUE are displayed to the user.

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. The application will check whether it has all the information to satisfy the expression stored locally and will perform the filter. 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.

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, dbblue, 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 'in' list. For example, for a Queue List the following are valid variables :-

queue	Queue Name
qtype	Queue type
Us	Usage value
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 0 to any characters
'?'	Matches 1 character.

As an example, suppose we have a list of queues displayed and wanted to limit those displayed by Queue Name. We could use the following

filters :-

queue == "A*"	To display only queues starting with 'A'.
queue == "SYSTEM*"	To display the system defined queues.
queue == "*C*"	To display only queues containing a 'C'.
queue == "*D"	To display only queues ending in a 'D'.
queue == "???"	To display queues with only three character names.

This wild card matching can be done with any string attribute of the list.

- 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 a 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

- 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.
- `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 the command string section of this manual.

Examples

On the basis that the best way to see what effect filters have is to 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 = "OS2"`
Show only queue "OS2"
- `queue == "OS2*"`
Show only queues with a name beginning "OS2"
- `queue == "*OS2"`
Show only queues with a name ending "OS2"
- `queue == "*OS2*"`
Show only queues with "OS2" 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.
- `bg(usage=xmitq,red)`
Show all queues but highlight the transmission queues
- `fg(qtype=local,red) & fg(qtype=remote,white)`
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.
- `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

Command Strings

A command string is a command you wish MQMON to issue to the operating system (OS/2 or Windows NT). 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 MQMON 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. My code 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 OS/2 and 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. I support the following characters :-

%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 is being issued not related to a list then it will be ignored and removed from the command string

Examples might be :-

"myprog.exe"

"myprog.exe -m OS2PGC1"

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

Problem determination

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 sign of this is that messages will build up on the remote Queue Managers Dead Letter Queue. My 'Q' program (SupportPac MA01) can be used to browse the message 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 OS/2 the userid will be OS2, for Windows NT it will be whatever the logged on userid. Therefore, the receiving Queue Manager, be it AS/400, MVS, 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.

Fixed problems

1. Running as a client to MVS hangs waiting for a reply.
2. Crash in Windows NT version if very large lists are displayed.
3. Failure to retrieve data from MVS if sub-system identifier not '='
4. Browsing queues on MVS running as a client to MVS gives command error.
5. Using a model queue fails to load configuration.

Known problems

1. Sometimes the OS/2 version crashes on startup
This happens very rarely, restart the application.
2. Painting problems when resizing OS/2 command windows
Resize the window slightly horizontally.

Migration from previous versions

Migration from a Version 4.0 or later

No changes are required.

Migration from a version prior to Version 4.0

Functionality

The general look and feel of Version 4.0 is virtually the same as previous versions however the internal workings of the code has been changed quite extensively. I have tried to ensure that all the old features of the program work as before and a number of customers have tested their set up for me but I can not guarantee that what used to work still does. I would recommend that you try out Version 4.0 before you delete any previous version you've been working on. If you find a problem with Version 4.0 please let me know and I'll do my best to provide a fix. The reason for the internal redesign was to support some new features which I'd hoped to incorporate into Version 4. Unfortunately I have not been able to spend as much time as I'd like on the code and since the code contains a number of changes I've been explicitly asked for I thought it worth releasing this new version now.

Main changes

The main differences from previous versions are :-

1. Main window can be displayed as a container/tree view
The Queue Manager objects are displayed if the location is expanded.
2. Loop-back monitoring
This means that it is no longer necessary to have user code at the monitored location.
3. Positions, Sizes, Filters and display options of dialogs can be saved
4. User can ask for confirmations of sensitive commands
5. Configuration is saved per Queue Manager
6. The application can run as a client
You can run it as a single thread to reduce connections
7. Configuration is automatically saved on application termination.

Configuration

Version 4.0 has 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, then end it. This will create a file with one Queue Manager entry.

You can then replicate the line to create the other destinations you need.

Configuration file

The configuration 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 := {ApplicationBlock}
ApplicationBlock :=
    Instance {QMGrBlock {DialogBlock} }
Instance :=
    "AP:" <QMGrName> <ReplyQ>
    [ 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
    [ 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 'tree' container view
    [ alarm ]
    Alarms are active
    [ monitor ]
    Monitoring is active
    [ confirms ]
    Show confirm dialogs
```

```

[ savepos ]
Save command dialog positions
[ savesize ]
Save command dialog sizes
[ font = <fontname> ]
Set window font name
[ fontsize = <fontsize> ]
Set window font size
{ clrn = <RGB number> }
Set area colour
QMGrBlock := "QM:" <QMGrName> <Location Description>
<Machine Type>
<Command Level>
[ mq = <Monitor Queue> ]
Set monitor queue name for location if not 'MQMON'
[ cq = <Command Queue> ]
Set command queue name for location if not normal value
[ mvs ]
Location is an MVS system
[ alarm ]
Alarms are active for this location
[ loop ]
Use loop-back monitoring
[ nc ]
No commands allowed
[ network ]
Location is part of the 'network'
[ disabled ]
Monitoring is disabled for this location
[ interval=<Monitor Interval> ]
Set monitoring interval, if not default
[ cmdok = <Command String> ]
Set command ok value fo this location
[ cmdfail = <Command String>]
Set command fail value for this location
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=<Refresh Rate> ]
Automatic reresh 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
[ ListBlock ]
[ FilterBlock ]
ListBlock := "LST:" <Number Attribute Pairs> { Attribute Pairs }
FilterBlock := "FLT:" { <FilterString> }
```