

# EVENTS MONITOR

## **Version 1.0**

15 July, 2002

Sanjeev. S  
India Software Labs  
IBM Global Services India Pvt. Ltd  
Bangalore

**Property of IBM**

**Take Note!**

Before using this report be sure to read the general information under "Notices".

**First Edition, March 2002**

This edition applies to Version 1.0 of *SupportPac title* and to all subsequent releases and modifications unless otherwise indicated in new editions.

© **Copyright International Business Machines Corporation 2001**. All rights reserved. Note to US Government Users -- Documentation related to restricted rights -- Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule contract with IBM Corp.

---

## Table of Contents

Notices .....	iv
Preface .....	vi
<b>Chapter 1. Introduction to MQSeries Instrumentation Events .....</b>	<b>1</b>
Event Types .....	1
<b>Chapter 2. Usage of Events Monitor .....</b>	<b>2</b>
Installation .....	2
Usage .....	3

---

## 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 be 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.

## ***Trademarks and service marks***

The following terms, used in this publication, are trademarks of the IBM Corporation in the United States or other countries or both:

- IBM
- MQSeries

---

## Summary of Amendments

### Date

15 July, 2002

### Changes

Initial release

---

## **Preface**

This tool (evmon) monitors the queue managers using events. MQSeries instrumentation events provide information about errors, warnings, and other significant occurrences in a queue manager. You can monitor the operation of queue manager by using 'evmon' tool.

It monitors all types of MQSeries events like Performance events, Channel events and Queue Manager events. Please read through to get more information.

---

## **Bibliography**

- *MQSeries System Administration*, IBM Corporation. SC33-1873-01.
- *MQSeries Event Monitoring*, IBM Corporation. SC34-5760-01.

---

## Chapter 1. Introduction to MQSeries Events

Instrumentation events cause special messages, called *event messages*, to be generated whenever the queue manager detects a predefined set of conditions. When the event occurs queue manager puts this message in the appropriate queue.

### Event Types

There are 3 types of instrumentation events, they are:

#### Performance Events

These events are notifications for the threshold conditions reached for the queue manager resources. For e.g.: queue depth high. When these events occurs queue manager puts the event messages into SYSTEM.ADMIN.PERFM.QUEUE.

#### Channel Events

These events are reported by channels as per their operations. For e.g.: an event is generated when a channel is started. These types of event messages will put into SYSTEM.ADMIN.CHANNEL.EVENT by the queue manager.

#### Queue Manager Events

These events are related to the definitions of resources in a queue manager. For e.g.: an event will be generated when an application tries to put a message into a queue, where PUT is disabled. These events are put into SYSTEM.ADMIN.QMGR.EVENT by the queue manager.

#### Note:-

1. We can define the event queues either as local queues or as local definition of remote queues.
2. Event message will not be available if the event queue is full.
3. The event become unavailable, if the event queue become full.



## **Chapter 2. Usage of Events Monitor(evmon)**

This tool (evmon) is useful for administrators and developers to monitor the queue managers with various types of events. It gives a formatted output for each types of events that are generated.

Event messages contain information relating to the origin of an event, including the type of event, the name of the application that caused the event and, for performance events, a short statistics summary for the queue.

Using evmon, you can monitor all the three types(qmgr events, channel events, and performance events) of event messages that MQSeries V5.2 provides. It is implemented by polling the event queues. There is an option available to set the frequency of polling. By default it will monitor all the event messages. There are options to choose the type of event that need to monitor. You can get the formatted event messages on your monitor, to a file, or both. If you choose the event messages to a file, then a file called evmon.log will be created to log the messages on your system.

It is also possible to monitor all the events from different queue managers. To achieve this, event queues should be local definition of remote queues. Remote queue name and remote queue manager name on these definitions should be the queue manager where evmon tool is running. Don't forget to create proper channels between these queue managers.

I have tested monitoring event messages of a different queue manager from a remote system. Tested this by monitoring event messages from an AIX sytem by running the evmon tool on my desktop having Windows NT. Following are the steps that I followed to do this:

AIX System - test qmgr is running.

Desktop(NT) - test1 qmgr is running.

### On AIX qmgr

1. Define a Sender channel to the desktop.

```
DEFINE CHL(CHL1) CHLTYPE(SDR) +
      CONNAME('<ip of the desktop(portno)>') +
      XMITQ('xq') +
      TRPTYPE(TCP)
```

2. Define an xmitq with name 'xq'

```
DEFINE QL('xq') USAGE(XMITQ)
```

3. Delete your default event queue.

```
DELETE QL(SYSTEM.ADMIN.QMGR.EVENT) PURGE
```

Note:- When you delete the event queue with above command you will Loose all the event messages that were existing, if any.

4. Define a remote queue name with same name of default event queue. Give remote queue name attribute as SYSTEM.ADMIN.QMGR.EVENT, and remote queue manager name as the queue manager in the system where you are running the evmon tool. In this case it is my desk top queue manager.

```
DEFINE QR(SYSTEM.ADMIN.QMGR.EVENT) +
      RNAME(SYSTEM.ADMIN.QMGR.EVENT)
      RQMNAME('test1') +
      XMITQ('xq')
```

#### On Desktop(Windows NT)

1. Create a receiver channel.

```
DEFINE CHL(CHL1) CHLTYPE(RCVR) TRPTYPE(TCP)
```

2. Run the listener

```
C:\>start runmqlsr -m test1 -t tcp -p 5500
```

3. Start the evmon tool. For eg:

```
evmon -m test1
```

Now all set to monitor the events from AIX system. Now you start the channel(CHL1) from AIX. Here I am redirecting only queue manager events to the desktop. You can do the same with all other event types also. Don't forget to enable the channel events on AIX queue manager. Here qmgr event messages on AIX queue manager(test) will be put into SYSTEM.ADMIN.QMGR.EVENT(defined it as a remote queue), and the messages will be redirected to desktop queue manager(test1 on NT).

## Installation

Download the programs to a directory and compile it on your system. Please follow the information's below to compile these programs on your system.

### On Solaris

```
cc -o libevperfm.so -G -K pic evperfm.c -lmqm
```

The above command will create a shared library called libevperfm.so. Copy this to /usr/lib and then give the following command.

```
cc -o evmon evmon.c -levperfm
```

This will create an executable called evmon

### On AIX

```
cc -o libevperfm.a -bexport:evperfm.exp -bM:SRE -bnoentry evperfm.c -lmqm
```

The above command will create a shared library. Copy this shared library to /usr/lib, then create an executable by linking it to main program, the command is as following:

```
cc -o evmon evmon.c -levperfm
```

### On HP

```
cc -c +z evperfm.c
```

Link the file to create a shared library, the command is:

```
ld -b -o libevperfm.sl evperfm.o
```

copy this shared library to /usr/lib, and then compile the main program to get the executable, the command is:

```
cc -o evmon evmon.c -levperfm -lmqm
```

### On Linux

Create a shared library

```
cc -shared -o libevperfm.so evperfm.c
```

Compile the above shared library with main program, the command is

```
cc -o evmon evmon.c -levperfm -lmqm
```

### On Winnt

```
cl /LD evperfm.c evperfm.def /link mqm.lib
```

The above command will create a DLL. To get the evmon executable, issue the following command:

```
cl -o evmon evmon.c evperfm /link mqm.lib
```

## Usage

With the exception of channel events, all instrumentation events must be enabled before they can be generated. You can enable and disable events by specifying the appropriate values for queue manager or queue attributes (or both) depending on the type of event. You can use MQSC commands to enable or disable the events.

### Enabling queue manager events

Enable the queue manager events by setting the appropriate queue manager attributes. Given below is an example for setting the queue manager attributes.

```
ALTER QMGR INHIBTEV(ENABLED)
```

### Enabling channel events

Most channel events are enabled automatically and cannot be enabled or disabled by command. The exceptions are the two automatic channel definition events. You can enable the automatic channel definition by giving the following command:

```
ALTER QMGR CHADEV(ENABLED)
```

### Enabling performance events

Performance events as a whole must be enabled on the queue manager, otherwise no performance events can occur. You can then enable the specific performance events by setting the appropriate queue attribute.

Following command is an example for setting the performance events.

```
ALTER QMGR PERFMEV(ENABLED)
```

```
ALTER QLOCAL('Q1') QSVCIINT(5000) QSVCIIEV(HIGH)
```

To start the event monitor on a queue manager called test, you need to give the following command:

```
evmon -m test
```

Note:- You must enable the events before giving this command.

Here are the options available with evmon:

- m - queue manager name from where you need to monitor event messages
- c - to monitor only channel events, by default all
- p - to monitor only performance events, by default all
- q - to monitor only qmgr events, by default all
- o - output to a file only, the output file is evmon.log
- s - output to stdout only by default the output will be to both
- w - waiting time on each type of event to happen, by default it will wait 10000 milliseconds on each queue. You can have an interval between 1 millisecond and 900000 milliseconds
- h/-? - Help

----- End of Document -----