Unit 6. Triggering concepts

What this unit is about

This unit gives a description of triggering in some detail.

There is a practical exercise on configuring WebSphere MQ for triggering and reply-to queues.

What you should be able to do

After completing this unit, you should be able to:

- Describe how triggering works
- Configure WebSphere MQ to enable triggering
- Determine the cause if triggering fails to occur

How you will check your progress

- Checkpoint questions
- Machine exercises
- Classroom discussion

References

SC34-6928 WebSphere MQ System Administration Guide

SC34-6941 WebSphere MQ Script (MQSC) Command Reference

SC34-6939 WebSphere MQ Application Programming Guide

SC34-6940 WebSphere MQ Application Programming Reference

Unit objectives

After completing this unit, you should be able to:
- Describe how triggering works
- Configure WebSphere MQ to enable triggering
- Determine the cause if triggering fails to occur

Figure 6-1. Unit objectives WM203 / VM2032.0

Notes:

One of the most common functions used in WebSphere MQ is triggering. It enables message-driven, or event-driven, processing to occur. During this topic, the requirements for triggering are discussed and you can gain a clear understanding of the advantages, as well as some of the more common pitfalls, associated with triggering.

Components of triggering

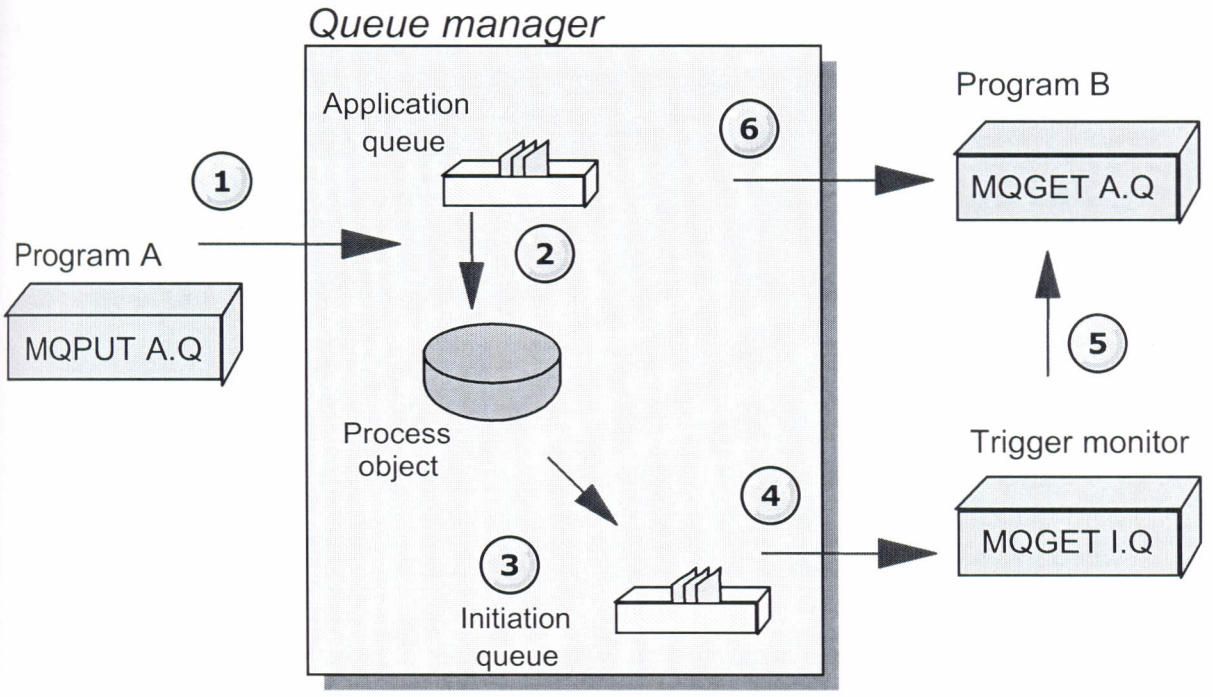

Figure 6-2. Components of triggering WM203 / VM2032.0

Notes:

The sequence of actions depicted on the visual are as follows:

1. Program A puts a message on an application queue which is enabled for triggering.
2. If the conditions for triggering are met, a *trigger event* occurs, and the queue manager examines the process object referenced by the application queue. The process object identifies the application to be started, Program B.
3. The queue manager creates a *trigger message* whose fields contain information copied from certain attributes of the process object and the application queue. The queue manager puts the trigger message on an *initiation queue*.
4. A long running program called a *trigger monitor* gets the trigger message, examines its contents, and starts Program B, passing the entire trigger message as a parameter.
5. Program B opens the application queue and gets messages from it.

Queue attributes controlling triggering

```
DEFINE  QLOCAL(MY_SERVER) TRIGGER  +
PROCESS(ECHO)  +
INITQ(SYSTEM.DEFAULT.INITIATION.QUEUE)
```

- TRIGGER
  NOTRIGGER
- TRIGMPRI(integer)
- TRIGTYPE(FIRST)
  TRIGTYPE(DEPTH)
  TRIGTYPE(EVERY)
  TRIGTYPE(NONE)
- TRIGDPTH(integer)
- TRIGDATA(string)

Figure 6-3. Queue attributes controlling triggering WM203 / VM2032.0

Notes:

The parameters of the WebSphere MQ command DEFINE QLOCAL which control triggering are:

- **TRIGGER -** Triggering is active
- **NOTRIGGER** - Triggering is not active
- **TRIGMPRI(*integer*) -** The threshold message priority for triggers. The queue manager ignores messages with a priority lower than this value when deciding whether a trigger event should occur.
- **TRIGTYPE(FIRST) -** A trigger event occurs when the queue changes from being empty to having one message on it.
- **TRIGTYPE(DEPTH)** - A trigger event occurs when the number of messages on the queue reaches the value indicated by the TRIGDPTH parameter.

  When triggering by depth, the queue manager disables triggering by setting the application queue to NOTRIGGER after it has created a trigger message. It is the responsibility of the application to reenable triggering by using the MQSET call.

- **TRIGTYPE(EVERY) -** A trigger event occurs for every message put on the application queue.
- ***TRIGDPTH(****integer****) -*** The number of messages that must be on the queue before a trigger event occurs for TRIGTYPE(DEPTH).
- **TRIGDATA(***string***) -** Data that is copied into the trigger message.

Process attributes

```
DEFINE  PROCESS(ECHO)  +
APPLICID('/opt/mqm/samp/bin/amqsech')
```

- APPLICID(*string*) Name of application to be started
- APPLTYPE(WINDOWS)
  APPLTYPE(UNIX)
- ENVRDATA(*string*) For use by trigger monitor
- USERDATA(*string*) For use by trigger monitor or the started application

Figure 6-4. Process attributes WM203 / VM2032.0

Notes:

A process and a queue are allowed to have the same name. The name of an object only has to be unique within an object type.

Other WebSphere MQ commands for processes are:

- ALTER PROCESS
- DELETE PROCESS
- DISPLAY PROCESS

The example in the figure is a UNIX example. It might not always be necessary to provide the full path as shown.

Triggering conditions

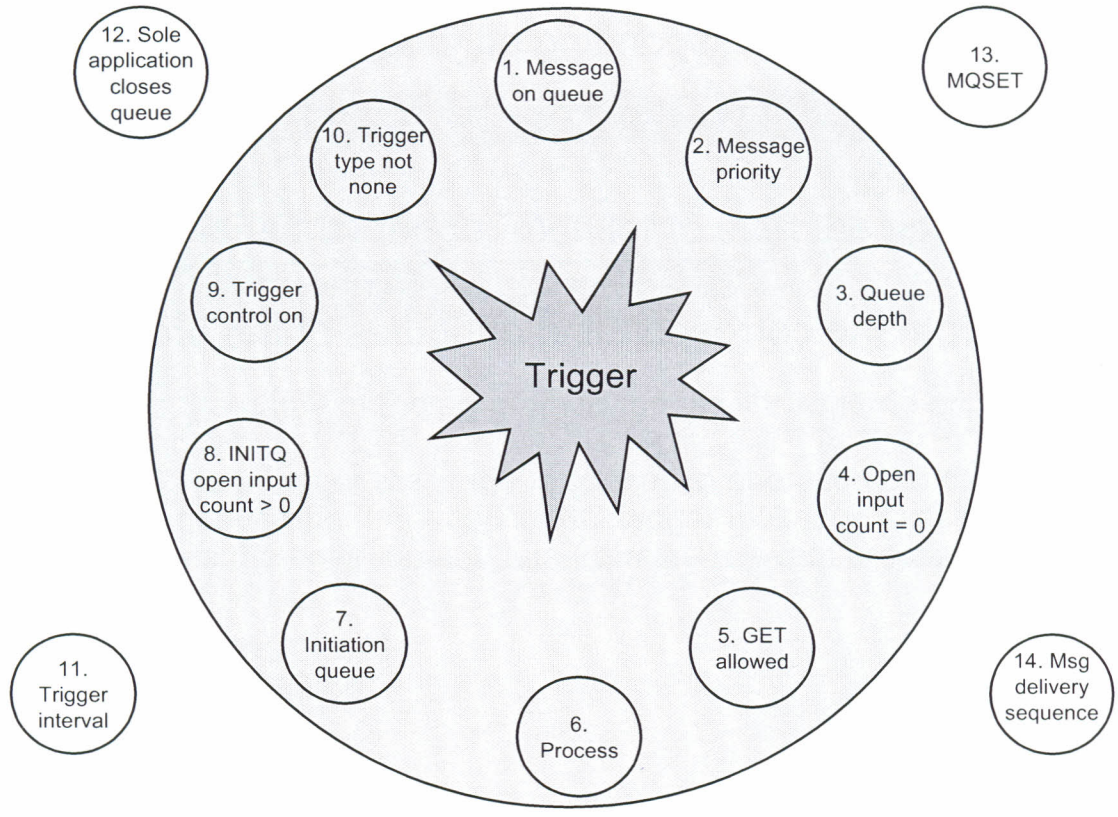

Figure 6-5. Triggering conditions WM203 / VM2032.0

Notes:

All of the conditions shown in the figure must be satisfied for a trigger event to occur.

If a trigger event does not occur when it is expected, check this list.

A full statement of all of these conditions can be found in the *WebSphere MQ Application Programming Guide.*

Fields in the trigger message

- Copied from the corresponding attributes of the application queue
  - QName
  - TriggerData
  - ProcessName
- Copied from the corresponding attributes of the process object
  - ApplType
  - ApplId
  - EnvData
  - UserData
- QMgrName

Figure 6-6. Fields in the trigger message

Notes:

The trigger message, structure MQTMC2, contains the fields:

- *QName* - The name of the application queue.
- *TriggerData* - The trigger data used by the started application. WebSphere MQ Versions 5 and 6 allow this field to be used to specify the name of the channel to be triggered.
- *ProcessName* - The name of the process object identifying the application that can service the application queue.
- *ApplType* - The application type. Examples are MQAT_CICS, MQAT_UNIX, and MQAT_WINDOWS.
- *ApplId* - The application identifier of the application to be started.
- *EnvData* - The environment data used by the trigger monitor.
- *UserData* - The user data for use by the trigger monitor or by the started application.
- *QMgrName* - The name of the queue manager that owns the application queue.

Trigger monitor

- To start a trigger monitor

```
runmqtrm -m QMgrName  -q InitiationQName
```

- Command issued by trigger monitor to start the application

```
<ApplId>  "<MQTMC2>"  <EnvData>
```

  - where
    - AppId is the name of the program to run, as it is entered on the command line
    - MQTMC2 is a trigger message, enclosed in quotation marks, from the initiation queue and formatted in MQ trigger message character format
    - EnvrData is the environment data from the relevant process definition

- Other trigger monitors
  - Client trigger monitor
  - CICS

Figure 6-7. Trigger monitor

Notes:

A number of trigger monitors are supplied with WebSphere MQ. The trigger monitor **runmqtrm** is supplied with all queue managers except WebSphere MQ for iSeries.

If the name of an initiation queue is not explicitly specified on the control command **runmqtrm**, the queue `SYSTEM.DEFAULT.INITIATION.QUEUE` is used by default.

The command which **runmqtrm** uses to start the application contains the structure MQTMC2 as a parameter. MQTMC2 represents *WebSphere MQ trigger message 2 (character format)*. This structure supplies the started application with the name of the queue that caused the trigger event and the name of the queue manager which owns it. The started application is therefore able to connect to the queue manager and open the queue.

Other trigger monitors are supplied by WebSphere MQ.

- On Compaq Open VMS Alpha, HP NonStop Kernel, UNIX systems, and Windows systems, there is a trigger monitor **runmqtmc** provided for WebSphere MQ clients. It links with the WebSphere MQ client libraries.

- On Compaq OpenVMS Alpha, HP NonStop Kernel, UNIX systems, and Windows use the standard trigger monitor, **runmqtrm**. The standard trigger monitor can be used for CICS transactions also.

It is possible to write your own trigger monitor to work in different ways to the ones supplied or to handle different application types.

Trigger monitor messages

- Messages are produced by:
  - Normal activities
    Examples:
    - Trigger monitor started
    - Waiting for a trigger message
    - Trigger monitor ended
  - Abnormal conditions
    Examples:
    - Initiation queue cannot be opened
    - Use of trigger monitor not authorized
    - Error starting triggered application
- Message can be written to:
  - Standard output device
  - Error log
- Trigger message may be written to dead letter queue
  Examples:
  - Trigger message structure is not valid
  - Trigger message specifies an unsupported application type
  - Trigger monitor is unable to start specified application

Figure 6-8. Trigger monitor messages WM203 / VM2032.0

Notes:

Messages relating to the operation of a trigger monitor are produced for two reasons.

- There are messages to report on normal activities such as when a trigger monitor starts and when it ends. These messages do not normally require any user action.
- There are messages to report on abnormal conditions such as when a trigger monitor fails to open the initiation queue or when it fails to start the specified application. These messages normally indicate that user action is required to correct the condition.

Checkpoint questions

1. What are the three trigger types?
2. What does the runmqtrm command do?
3. Which of the following messages from the trigger monitor are reporting on normal activities?
   a. Waiting for a trigger message
   b. Initiation queue cannot be opened
   c. Use of trigger monitor not authorized
   d. Trigger monitor ended

4. Which of the following conditions are essential for TRIGTYPE(FIRST) to trigger:
   a. Queue Depth goes from 0 to 1
   b. OPPROCS = 0
   c. IPPROCS = 0
   d. Trigger monitor must be running against the relevant INITQ

Figure 6-9. Checkpoint questions WM203 / VM2032.0

Notes:

Unit summary

Having completed this unit, you should be able to:
- Describe how triggering works
- Configure WebSphere MQ to enable triggering
- Determine the cause if triggering fails to occur

Figure 6-10. Unit summary WM203 / VM2032.0

Notes:

Use of message-driven processing enables the starting of applications as they are needed, alleviating the requirement to manually start them. This is accomplished through the use of triggering.