The IBM ATS WebSphere MQ team:

Lyn Elkins – elkinsc@us.ibm.com
Mitch Johnson – mitchj@us.ibm.com
Eugene Kuehlthau - ekuehlth@us.ibm.com
Ed Zeilnhofer - edz@us.ibm.com
Introduction

The document describes the sample COBOL MQ CICS transactions called QPUB. This transaction uses the publication feature of WMQ V7 and requires CICS 3.2 or CICS 4.1.

Please note that the following PTFs need to be applied to support this transaction

CICS TS 3.2 – PK66866 (UK52671,UK52672,UK52673,UK52680) OR
CICS TS 4.1 – PK89844 (UK52619,UK52667,UK52668,UK52669)

This transaction is single purpose – all it does it publish a number of messages to the topic and/or topic string provided in the control message. The control message, the message used to start the transaction, and fields from the process definitions are used to provide the necessary information to create the publications and to report on the status of the publication request.

This document assumes the reader is somewhat familiar with WMQ, CICS, and COBOL.

Terms

Control Message – the message used to start the QPUB transaction. It contains the basic information about the number of publication, the MQ Topic object and the MQ Topic string.

Resolved topic – the resolved topic is the complete topic string used for publications. It can be made up of the topic string parameter from the defined topic object, the topic string supplied in the control message when no topic object has been specified, or a concatenation of the topic string attribute of the topic object followed by a slash ('/') and the topic string from the control message. In the sample program, the resolved topic is limited to 400 characters.

Trigger message – this is the message passed to the processing program when the transaction is triggered. A complete description of the trigger message is documented in the WebSphere MQ InfoCenter, or for those have the older MQ manuals in the Application Programming Reference (publication number SC34-6940).

Note

Please note that the QPUB transaction was introduced in the Redbook, High Availability in WebSphere Messaging Solutions, publication number SG24-7839-00. This version of the QPUB transaction is slightly different, it has been expanded to include both the topic object and topic string parameters.
QPUB – Sample CICS transaction to publish messages

Description:

The QPUB transaction executes the QPUBCBL program, it publishes messages to a topic object, topic string or a combination of the two fields. It is started by a formatted ‘control message’, which triggers the transaction. It also can use information from the WMQ process definition.

Inputs:

The Publish Control message, which has the following layout:

```
01 PUB-CONTROL-MESSAGE.
   05 NUMBER-OF-PUBS     PIC 9(5)       VALUE ZEROS.
   05 TOPIC-OBJECT            PIC X(48)    VALUE SPACES.
   05 TOPIC-STRING            PIC X(200)  VALUE SPACES.
```

The fields are used as follows:

- **NUMBER-OF-PUBS** is the number of messages to be published to the resolved topic, this can be a range of 00001-99999.
- **TOPIC-OBJECT** is the MQ defined topic object. The field is optional, if omitted the TOPIC-STRING value is used as the resolved topic. If supplied, the topic string from the topic object definition is used as part (or the whole of) the resolved topic.
- **TOPIC-STRING** is an optional component of the resolved topic. If omitted and the TOPIC-OBJECT is also omitted, the SYSTEM.DEFAULT.TOPIC is used for publication. If it is present, and the TOPIC-OBJECT is omitted, this value is used as the resolved topic for publication. If it is present and the TOPIC-OBJECT is also present, the resolved topic is created by WMQ from the topic string defined on the topic object, a concatenated slash to indicate hierarchy, and the topic string.

The following table describes the resolved topic that will be used for message publication. It assumes that the topic object named has a topic string that contains the name, though that is not a requirement.

<table>
<thead>
<tr>
<th>TOPIC-OBJECT</th>
<th>TOPIC-STRING</th>
<th>Resolved topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blank</td>
<td>Blank</td>
<td>‘SYSTEM.DEFAULT’TOPIC’</td>
</tr>
<tr>
<td>Fruit</td>
<td>Blank</td>
<td>Fruit</td>
</tr>
<tr>
<td>Cookware</td>
<td>Castiron/Skillet</td>
<td>Cookware/Castiron/Skillet</td>
</tr>
<tr>
<td>Blank</td>
<td>ShoeSale/Sneakers</td>
<td>ShoeSale/Sneakers</td>
</tr>
</tbody>
</table>
The Trigger message contains data fields that are used as follows:

- MQTM-QNAME – the name of the publication control queue, in the sample delivered it is ‘QPUB.CONTROL.QUEUE’
- MQTM-ENVDATA – this is taken from the process definition, and may be used to supply the status queue name (see Outputs). If not supplied on the process definition, this defaults to ‘QPUB.STATUS.QUEUE’.
- MQTM-USERDATA – this is taken from the process definition and may be used to supply part of the publication message body (see Outputs). If not supplied on the process definition it defaults to 'NO USER DATA PROVIDED'.
 Outputs:

The published messages, which have the following layout:

```
01 PUBLISH-MESSAGE.
   05 PM-USER-DATA   PIC X(100) VALUE SPACES.
   05 PM-NUMBER     PIC 9(5)   VALUE ZEROS.
   05 PM-TOPIC      PIC X(48)  VALUE SPACES.
   05 PM-RESOLVED   PIC X(400) VALUE SPACES.
```

The fields are used as follows:

PM-USER-DATA is the value from the process definitions or the ‘NO USER DATA PROVIDED’.
PM-NUMBER – this is the number of the publication. All publications have a sequential number starting at 1 and going thru 999.
PM-TOPIC – the topic object, if provided, from the publish control message.
PM-RESOLVED – this is the resolved topic string used for the publication.

The status message, which has the following layout:

```
01 STATUS-MESSAGE.
   05 FILLER       PIC X(20)
                  VALUE 'PUBLICATIONS MADE= '.
   05 SM-NUMBER    PIC 999 VALUE ZEROS.
   05 FILLER       PIC X(20)
                  VALUE ' FOR TOPIC = '.
   05 SM-TOPIC     PIC X(48) VALUE SPACES.
   05 FILLER       PIC X(20)
                  VALUE ' RESOLVED = '.
   05 SM-RESOLVED  PIC X(400) VALUE SPACES.
```

The fields are used as follows:

SM-NUMBER is the total number of messages published.
SM-TOPIC – the topic object, if provided, from the publish control message.
SM-RESOLVED – this is the resolved topic string used for the publication.
QPUB – Sample COBOL WMQ CICS Publish Transaction

**QPUBCBL Program Flow:**

1) The QPUB transaction is triggered.

2) The control queue is opened.

3) Publication control message is read.

4) The publication is built from the data provided in the message and the process definition.

5) The topic is opened.

6) Messages are published in a loop, until the publication count has been reached.

7) The status message is built.

8) The status queue opened and the status message is put.

9) The topic and all queues are closed.

10) Control returns to CICS.

If testing with a resolved topic string of longer then 400 characters is required, the size can be adjusted in the program QPUBCBL and the program compiled and linked. The following fields may need to be altered to support a larger area:

- **TOPIC-STR** – 200 byte working storage field that has a copy of the input topic string from the publication control message.
- **RESOLVED-TOPIC-STR** – 400 byte working storage field that is used to pass to WMQ to construct the resolved topic string from the topic object and topic string from the publication control message.
- **TOPIC-STRING** – 200 byte working storage field that is part of the publication control message.
- **PM-RESOLVED** – 400 byte working storage field that is part of the published message, it gives the resolved topic string returned from WMQ.
- **SM-RESOLVED** - 400 byte working storage field that is part of the status message, it gives the resolved topic string returned from WMQ
CICS Definitions:

There are two CICS definitions required, one for the QPUB transaction and one for the QPUBCBL COBOL program. If the names need to be altered to comply with naming standards, the source for the COBOL program QPUBCBL and the CICS definitions are included.

QPUB definition:

Using CEDA the QPUB definition was created using the following steps:

1) Enter the CEDA defining the transaction and it’s program to a specified group, in the example the group QML0GRP is used.

CEDA DEF TRANS(QPUB) PROGRAM(QPUBCBL) GROUP(QML0GRP) _
2) The definition success message should appear as shown. Note that the TASKDATALoc can be Any, in this example it was allowed to default.
QPUBCBL Definition:

1) Enter the define program command, supplying the group name.

CEDA DEF PROG(QPUBCBL) GR(QML0GRP)

2) Set the language to COBOL, the data location to any and hit the enter key. The ‘Define Successful’ message should be displayed as follows.

QML0GRP Installation

The new resources must be installed into the region.

1) Enter the CEDA install command as shown:

CEDA IN G(QML0GRP)
2) The install successful messages should be displayed as shown:

```
IN G(QMLOGRP)
OVERTYPE TO MODIFY
CEDA Install
  ALL
    ATomservice ==> -
    Bundle ==> -
    CONNnection ==> -
    CORbaserver ==> -
    DB2Conn ==> -
    DB2Entry ==> -
    DB2Tran ==> -
    DJar ==> -
    DOctemplate ==> -
    Enqmodel ==> -
    File ==> -
    Ipconn ==> -
    J0urnalmodel ==> -
    JVMserver ==> -
    LIBrary ==> -
  + LSrpool ==> -

        SYSID=T0R1 APPLID=CTSTOR01

INSTALL SUCCESSFUL       TIME: 12.00.21   DATE: 01/20/11
```
**WMQ Definitions:**

**Process Definition, QPUB.PROCESS**

To use triggering to initiate the QPUB transaction, a process definition is needed. This definition includes the CICS transaction name, if QPUB is not used as the transaction name this needs to be altered to include that. The definition also includes User and Environment data the program QPUBCBL uses to supply information about the publication message and the status queue. If the supplied values are not correct for the environment being used, please alter the values as necessary.

This process definition sample is included with the TechDoc as QPUB_PROCESS_DEF.txt. From MQ Explorer the process definition is shown below.
Queue Definitions

Three queues are needed for testing the QPUB transaction. The samples are defined as follows:

- **QPUB.CONTROL.QUEUE** – the publication control queue. This queue is triggered and initiates the publication process. If changes have been made to the CICS transaction name or if the CICS initiation queue for the region where the QPUB transaction will run is not the default CICS01.INITQ, those values need to be changed before defining this queue.
- **QPUB.STATUS.QUEUE** – this publication status queue. This queue holds status messages, generated after all publications have been made.
- **QPUB.SUB.QUEUE** – the subscription queue. This queue is the subscription queue, used for testing that the publication process is working.

Sample queue definitions are included with this TechDoc as QPUB_QUEUE_DEF.txt. A display of the MQ Explorer definitions of each queue and the necessary attributes for this sample are shown on the following pages. Note that the required fields are listed in a table below the picture, the other fields have default values.
1) QPUB.CONTROL.QUEUE

The basic definition:

Required Fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Queue Name</td>
<td>QPUB.CONTROL.QUEUE</td>
</tr>
</tbody>
</table>
The triggering information for QPUB.CONTROL.QUEUE:

Required Fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trigger control</td>
<td>On</td>
</tr>
<tr>
<td>Trigger type</td>
<td>First</td>
</tr>
<tr>
<td>Trigger depth</td>
<td>1</td>
</tr>
<tr>
<td>Trigger message priority</td>
<td>0</td>
</tr>
<tr>
<td>Initialization queue</td>
<td>CICS01.INITQ</td>
</tr>
<tr>
<td>Process name</td>
<td>QPUB.PROCESS</td>
</tr>
</tbody>
</table>
2) QPUB.STATUS.QUEUE

Required Fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Queue Name</td>
<td>QPUB.STATUS.QUEUE</td>
</tr>
</tbody>
</table>
3) QPUB.SUB.QUEUE

4) Required Fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Queue Name</td>
<td>QPUB.SUB.QUEUE</td>
</tr>
</tbody>
</table>
### Topic Object Definition, QPUB.TEST

A sample topic definition QPUB.TEST is delivered. The topic string associated with this object is ‘QPUB’.

This sample is included with the TechDoc as QPUB_TOPIC_DEF.txt.

From the MQ Explorer the topic object looks as shown below.
5) Required Fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topic Name</td>
<td>QPUB.TEST</td>
</tr>
<tr>
<td>Topic string</td>
<td>QPUB</td>
</tr>
<tr>
<td>Description (optional)</td>
<td>QPUB TRANSACTION SAMPLE TOPIC</td>
</tr>
</tbody>
</table>
Subscription Definition

Two sample administrated subscription definitions for testing are delivered as part of this TechDoc. The first, QPUB.TESTSUB1 subscribes to the topic object only (QPUB.TEST). The second, QPUB.TESTSUB2 subscribes to the combination topic object (QPUB.TEST) and topic string value TEST.

The sample included with the TechDoc as QPUB_SUBS_DEF.txt. Note that the second definition includes the destination queue manager name, which is optional on the subscription definition.
From an MQ Explorer the definitions would look as follows:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subscription Name</td>
<td>QPUB.TESTSUB1</td>
</tr>
<tr>
<td>Topic name</td>
<td>QPUB.TEST</td>
</tr>
<tr>
<td>Destination name</td>
<td>QPUB.SUB.QUEUE</td>
</tr>
</tbody>
</table>

Note: the Topic string is shown on the display, but has been pulled from the topic object definition. It should not be entered if defining the subscription using the explorer.
Required Fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subscription Name</td>
<td>QPUB.TESTSUB2</td>
</tr>
<tr>
<td>Topic Name</td>
<td>QPUB.TEST</td>
</tr>
<tr>
<td>Topic string*</td>
<td>TEST</td>
</tr>
<tr>
<td>Destination queue manager -</td>
<td>Your queue manager name</td>
</tr>
<tr>
<td>options</td>
<td></td>
</tr>
<tr>
<td>Destination name</td>
<td>QPUB.SUB.QUEUE</td>
</tr>
</tbody>
</table>

Note: the Topic string shown on the display has been automatically created by WMQ by concatenating the string attribute of the topic object, a slash (‘/’) and the string entered, but has been pulled from the topic object definition. It should not be entered if defining the subscription using the explorer. * DO NOT enter ‘QPUB/TEST’ or ‘/TEST’ if defining the subscription via the MQ Explorer. Enter ‘TEST’ only.
**Testing the QPUB Sample:**

Two sample publication control messages are supplied. The first one tests publication to a topic object and the sample publication control message is delivered in QPUB_TEST1.txt. The second one tests publication to a topic object and string combination, and the sample control message is delivered in QPUB_TEST2.txt.

**Test 1 – Topic object only specified**

For this test, the MQ Explorer was used to put the sample publication control message and to review the output.

1) Using the MQ Explorer to queue definitions and their current depth are checked:

![Queues](image1.png)

2) Right click on the QPUB.CONTROL.QUEUE and select ‘Put test message’
3) The blank input panel should be displayed as shown

![Put test message](image2.png)
4) Copy the contents of QPUB_TEST1.txt into the ‘Message data’ field, as shown. Click on the ‘Put message’ button.

5) Close the input panel.
6) Refresh the queue list, you should see counts like those shown:

    | Queue name           | Queue type | QSG dispos... | Open input count | Open outp... | Current queue depth |
    |----------------------|------------|----------------|-------------------|--------------|---------------------|
    | QPUB.CONTROL.QUEUE   | Local      | Queue man...   | 0                 | 0            | 0                   |
    | QPUB.STATUS.QUEUE    | Local      | Queue man...   | 0                 | 0            | 1                   |
    | QPUB.SUB.QUEUE       | Local      | Queue man...   | 0                 | 0            | 2                   |

7) The status queue should contain a message that looks like this (use the browse messages option from MQ Explorer):

<table>
<thead>
<tr>
<th>Position</th>
<th>Put date/time</th>
<th>User identifier</th>
<th>Put application name</th>
<th>Format</th>
<th>Date length</th>
<th>Message data</th>
<th>Accounting</th>
<th>Message type</th>
<th>QPUB.TEST</th>
<th>RESV/EX</th>
<th>390/46420X</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Dec 8, 2010 6:11:00 AM</td>
<td>STCBAF</td>
<td>CTST01/QPUB</td>
<td>MPRTR 51</td>
<td>PUBLISHED MADE= 002 FOR TOPIC = QPUB.TEST</td>
<td>RESV/EX = 390/46420X</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
8) Right click on the message and select properties and click on the data folder to view the message contents. The message body should look as is shown:

9) Click on ‘Cancel’ and return to the queue list.
Test 2 – Topic object and string specified

1) Right click on the QPUB.CONTROL.QUEUE and select ‘Put test message’
2) The blank input panel should be displayed, as shown above. Copy the sample message from QPUB.TEST2.txt into the message field (as shown) and hit the ‘Put message’ button. Then close the panel.

3) Refresh the queue list, the current depths should look as follows:

```
+-------------------------------------------------+
<table>
<thead>
<tr>
<th>Queue name</th>
<th>Queue type</th>
<th>QSG disposition</th>
<th>Open input count</th>
<th>Open output count</th>
<th>Current queue depth</th>
</tr>
</thead>
<tbody>
<tr>
<td>QPUB.CONTROL.QUEUE</td>
<td>Local</td>
<td>Queue manager</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>QPUB.STATUS.QUEUE</td>
<td>Local</td>
<td>Queue manager</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>QPUB.SUB.QUEUE</td>
<td>Local</td>
<td>Queue manager</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
</tbody>
</table>
+-------------------------------------------------+```
4) Browse the QPUB.PUB.STATUS queue, and open the properties for the second message on the queue, and open the data folder. Examine the data, the resolved topic should be ‘QPUB/TEST’.

![Message 2 - Properties](image)

5) You have successfully implemented and tested the QPUB sample program, demonstrating the use of publications for CICS COBOL programs.
Acknowledgements:

The authors would like to thank the following people for their assistance and testing:

Mark Taylor
Shalawn King
Jenifer Foley
Chris Griego
Ashley Curry