73

MQ update

July 2005

In this issue

A itc PUBLICATION

MQ Update

Published by

Xephon Inc
PO Box 550547
Dallas, Texas 75355
USA

Phone: 214-340-5690
Fax: 214-341-7081

Editor

Trevor Eddolls
E-mail: trevore@xephon.com

Publisher

Colin Smith
E-mail: info@xephon.com

Disclaimer

Readers are cautioned that, although the information in this journal is presented in good faith, neither Xephon nor the organizations or individuals that supplied information in this journal give any warranty or make any representations as to the accuracy of the material it contains. Neither Xephon nor the contributing organizations or individuals accept any liability of any kind howsoever arising out of the use of such material. Readers should satisfy themselves as to the correctness and relevance to their circumstances of all advice, information, code, JCL, scripts, and other contents of this journal before making any use of it.

Subscriptions and back-issues

A year's subscription to *MQ Update*, comprising twelve monthly issues, costs $380.00 in the USA and Canada; £255.00 in the UK; £261.00 in Europe; £267.00 in Australasia and Japan; and £265.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the July 2000 issue, are available separately to subscribers for $33.75 (£22.50) each including postage.

Contributions

When Xephon is given copyright, articles published in *MQ Update* are paid for at the rate of $160 (£100 outside North America) per 1000 words and $80 (£50) per 100 lines of code for the first 200 lines of original material. The remaining code is paid for at the rate of $32 (£20) per 100 lines. To find out more about contributing an article, without any obligation, please download a copy of our *Notes for Contributors* from www.xephon.com/nfc.

MQ Update on-line

Code from *MQ Update*, and complete issues in Acrobat PDF format, can be downloaded from our Web site at www.xephon.com/mq; you will need to supply a word from the printed issue.

Printed in England.

WebSphere MQ Version 6.0 – why it's important

The recently-announced Version 6.0 of WebSphere MQ contains, IBM claims, in excess of 150 enhancements, amongst which can be found improved support for building and using Enterprise Service Buses (ESBs).

"What's an ESB?" you ask. Well, earlier in the year IBM was describing it as a way of connecting everything to everything. An ESB makes use of messaging technology plus Service-Oriented Architecture (SOA), XML, Web services protocols, and intelligent routing to allow disparate systems to be linked together. So that's all the fashionable acronyms in one product!

WebSphere MQ already includes a large library of connectors to Oracle, SAP, and Siebel systems applications, as well as mainframe systems such as CICS and IMS.

It's been over two years since WebSphere was last upgraded – and IBM has made available nine support packs since then. Those packs have needed to increase the amount of messaging and length of messages the software supports, so in WebSphere 6.0 IBM has increased the size of the queues in which messages can be stored from 1GB to 4GB.

Although IBM claims to have 12,000 customers, it knows that there are a lot of sites out there that could be using WebSphere – and it's looking to get their business. So if they're not using WebSphere, what are they doing? Figures being bandied about suggest that around three-quarters of middleware users are using little more than FTP plus a bit of home-grown code to move data around the enterprise. And, although you might suggest that they would say that wouldn't they, IBM is claiming that this low-level technology is beginning to show clear signs of not being enough.

There are other pressures that these companies are facing. For example, there is a lot of talk about SOA and other modern approaches to getting the most out of legacy applications.

In addition, there are regulatory compliance issues (we're talking Sarbanes-Oxley and HIPAA). And thirdly there is the difficulty of supply-chain integration. Companies are realizing that they need to integrate their internal data in order to be able to provide the accurate data quickly to their suppliers.

But what happens at sites where FTPing and custom-code seem to be working at the moment? What's IBM's master plan to get their business? Cunningly, IBM has incorporated a file transfer application that allows customers to continue working in much the same way. Customers who like their home-grown scripts can make use of a new utility that allows the scripts to be incorporated into WMQ. The utility supports a scripting language designed to help customers migrate their scripts to WMQ and take advantage of WMQ's functions.

The users know what they are doing because it's similar, but the advantages of WMQ are added on – WebSphere MQ will make sure that any intended recipient is ready to receive the file and it will also confirm that the message has been received after it has been sent. And your friendly neighbourhood IBM salesman will be glad to spell out the benefits of logging each activity, assured delivery, and MQ's monitoring and configuration tools. It's a manageable, auditable, and reliable message-queueing and data-integration framework – they'll say.

Also new is an Explorer console that's derived from the Eclipse IDE framework. What makes it special is that it runs under Linux or Windows and allows users to control their entire MQ infrastructure – and that includes z/OS and iSeries.

Having an interface to the Eclipse open source programmer's workbench means that the many Eclipse-based development and testing tools are now available for the development of applications that can use the WMQ message bus for communication with the other parts of a company's IT infrastructure. This makes it easier for WMQ users to build composite applications or make use of a service-oriented architecture.

What else is included in those 150 enhancements? Developers who like working on Windows will be pleased with integrated support for .NET classes. This allows .NET applications to natively invoke MQ. There's also SOAP support (another must-have acronym). WebSphere Version 6.0 can send and receive Simple Object Access Protocol (SOAP) messages, which are often used in establishing Web services between applications running on disparate systems. Previously, WebSphere MQ could only read SOAP messages; now SOAP-based messaging is tightly integrated.

Enhancements also include 64-bit interfaces on AIX, HP-UX, and Solaris. The updated version also includes platform support for Linux for iSeries and Linux for pSeries.

And, when used with WebSphere Application Server (WAS) and WebSphere Business Integration (WBI), there's ESB capability – or did I mention that earlier?

Nick Nourse
Independent Consultant (UK)

Writing a self-subscribing input node for WebSphere Business Integration Message Broker

This article describes a way to implement an input node that subscribes itself to publications in a WebSphere Business Integration Message Broker (hereafter called WebSphere BI MB) environment.

PUBLISH/SUBSCRIBE COMMUNICATION

In general, there are different communication methods available when exchanging information between applications. The most commonly used one is a request/response pattern, where one application is sending a request to a second

application and this returns information in a response. Such communication is used in most client/server applications. It needs the first application to be aware of the second one and know how to reach the application.

A second pattern is one-way, or fire-and-forget, communication. In this style, an application sends some information and does not get back any indication whether or not the information has been processed. Such communication is often used with WebSphere MQ, where a WebSphere MQ queue manager can be certain that the message is delivered.

A third communication method is the publish-and-subscribe pattern. In this method, the sending and the receiving applications are decoupled. There is one application providing information about a specific topic – providing the information is called publishing. There can be zero, one, or any other number of applications that are interested in information about the topic. Registering an interest in a topic is called a subscription. The publishing application usually doesn't know whether or not there are applications requesting the information. Therefore there must be a component that mediates between the publishing and the subscribing applications. Such a component is called a message broker.

MESSAGE PROCESSING IN WEBSPHERE BI MB

WebSphere BI MB is a message broker that supports all kinds of communication. It allows users to write message flows that process or send messages. It is also a full-function message broker.

MESSAGE PROCESSING THE MESSAGE BROKER

A message in WebSphere BI MB can be processed using a message flow. A message flow is a definition of actions to a message. The actions are called nodes. The way that the nodes are wired determines how the message is processed. A message flow always starts with an input node. There are

different kinds of input node already available with the broker – for example an MQInput node can be used to read messages from a WebSphere MQ queue. Any user can define additional input nodes.

If a message is received by the input node, the message broker starts processing it. As part of the message flow, other messages, either responses to the actual message or new request messages, can be generated.

PUBLISH/SUBSCRIBE IN THE MESSAGE BROKER

WebSphere BI MB can also be used by publish/subscribe applications. Publishing is not done directly; instead, a publisher sends a message to a message flow. One of the provided nodes of the message broker is a publication node. If the message being processed reaches such a publication node in the message flow, it triggers the distribution of the information to the subscriber. The broker checks for subscriptions to the associated topic for the current message and sends out information to the subscribers.

In contrast to publishing, a subscriber must directly register its interest in information about a specific topic by sending a message to the broker. Such subscription requests are handled by a WebSphere BI MB internal message flow. Part of the subscription request is the topic and the destination where the subscriber waits for the information. Such a destination is in many cases a WebSphere MQ message queue.

SUBSCRIBING MESSAGE FLOWS TO A TOPIC

To issue a subscription request message to the message broker a program must be started. It can then actively send a message with its selection criteria to the message broker. Unfortunately, a normal message flow can’t issue a subscription. The main reason is that the message flow does not become active until the first message arrives. It never gets a message because the input queue of the message flow has

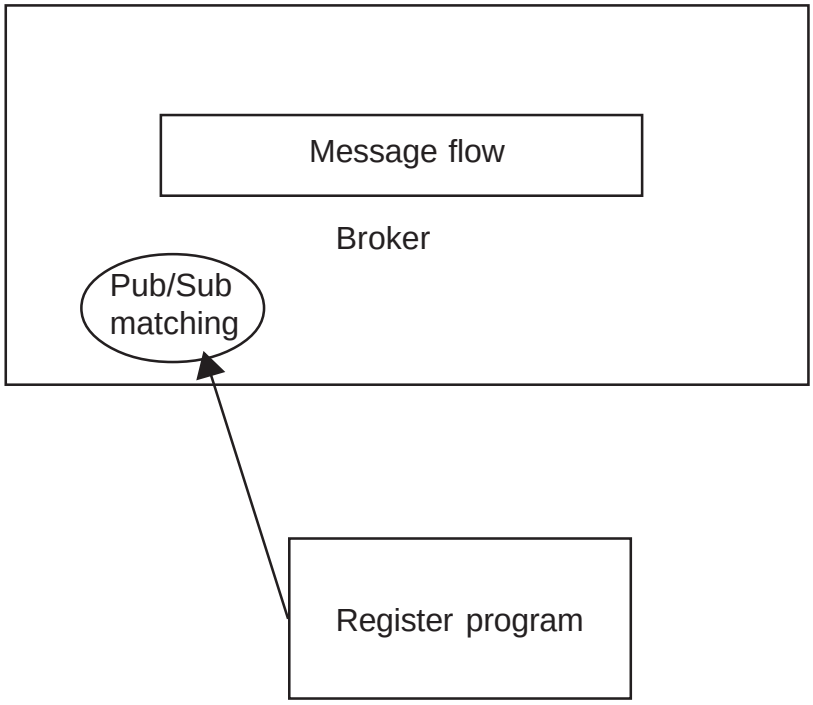

Figure 1: Registering a subscription by an external program

not subscribed to the particular topic – because it is not yet active.

The standard solution for this problem is to provide an external program that initiates the subscription on behalf of the message flow, as shown in Figure 1. The external program sends a subscription request to the message broker and registers the message flow. The subscription request specifies the input queue name of the message flow as the destination for the publications. The response to the subscription request is handled by the external program. It can check whether subscribing to the requested topic was successful. Once the subscription is complete, the message broker sends all publications about the selected topic to the message flow's input queue.

This solution has some issues. The main problem can result from the situation where at installation time of the message

flow, the step to run the external program is not performed and therefore there is no subscription. Other problems can occur, for example, if the issuing of the subscription is not successful and this is not detected. It is also possible that the subscription disappears after some time. There are some situations where the broker itself cancels a subscription, for example, if the information cannot be delivered. It may also happen that the subscription is inadvertently deleted by a WebSphere BI MB publish and subscribe administrator. Therefore it is necessary to check the existence of the subscription every so often.

Another drawback with this external program can become apparent if topic security is activated by running the broker domain with a User Name Server. The external program potentially runs with a different user ID from the message flow. This may lead to inconsistencies in the security settings; for example, it may happen that the user of the external program is not allowed to subscribe to the specific topic while the message owner (the user ID of the message broker) is allowed. The user of the external program must then be allowed access to the topic. Another possibility is to run the program with the user ID that is also used to run the message broker. Therefore someone must log on to the system with this user ID and run the program. Since a message broker user ID is usually a functional user ID not suited to logon, this requires a lot of administration.

A SELF-SUBSCRIBING MESSAGE FLOW

The drawbacks described in the previous sections are not acceptable in some situations. One such situation arose for WebSphere Business Integration for Financial Networks (abbreviated to WebSphere BI for FN) – an IBM software product. This product exploits the message broker. It consists of a base product and extensions that provide access to different financial networks. For some accounting services in the product, it is necessary to publish message broker statistics and accounting information and process a message flow. For further details about WebSphere BI for FN see www.ibm.com/software/integration/wbifn.

Relying on correct installation actions and regularly checking that the subscriptions still exist is not appropriate. In situations where the subscriptions are not processed, the usage of the product is not correctly accounted for. To avoid these situations, different methods of allowing a message flow to subscribe itself to a topic have been evaluated.

The first and most obvious method is to write its own input node as a plug-in input node. This would allow all the processing needed to send the subscription message and check the results to be coded. The node could regularly check that the subscription still exists. The main drawback with this method is that it duplicates a lot of the message broker that already exists, for example the MQInput node processing, constructing and parsing MQRFH2, and XML data structures. To avoid this, another method has been developed.

THE MAIN IDEA

For a message flow, the main problem with subscribing to a topic is that it doesn't become active until a message is received that triggers processing. This problem has been bypassed by letting the message flow subscribe itself to the requested topic. The processing for the subscription is triggered by another input node that is already available with WebSphere BI for FN Base. This node is a timer input node; it regularly issues timer input messages. The interval between issuing the messages is a property of the input node. Timer messages are simple WebSphere BI for FN messages consisting of only MQMD and MQRFH2.

Every time the timer input node issues a message to its out terminal, the timer input messages are used to trigger subscription processing. Within the processing path that follows the out terminal, a subscription request message is generated and sent to subscribe the current message flow to the requested topic. Therefore the name of the WebSphere MQ queue name for the main processing in the message flow is passed as a destination for the publications. The

subscriptions can be processed with varying degrees of complexity. For example, it is possible simply to issue the subscription message to the broker, or there could be another MQInput node in the subscription processing that receives the response to the subscription request, analyses the completion information and issues messages in the case of errors. The level of complexity depends on the amount of recoverability needed.

To build the subscription request message, send it to the broker, and analyse the response, standard WebSphere BI MB nodes (such as Compute and MQOutput nodes) can be used.

The timer input node has a property that determines the interval between issuing the timer input messages. The first message is always issued directly after the message flow is started. How to set this property needs to be carefully considered. In normal situations it is sufficient to process just the initial timer input messages; then no further messages are

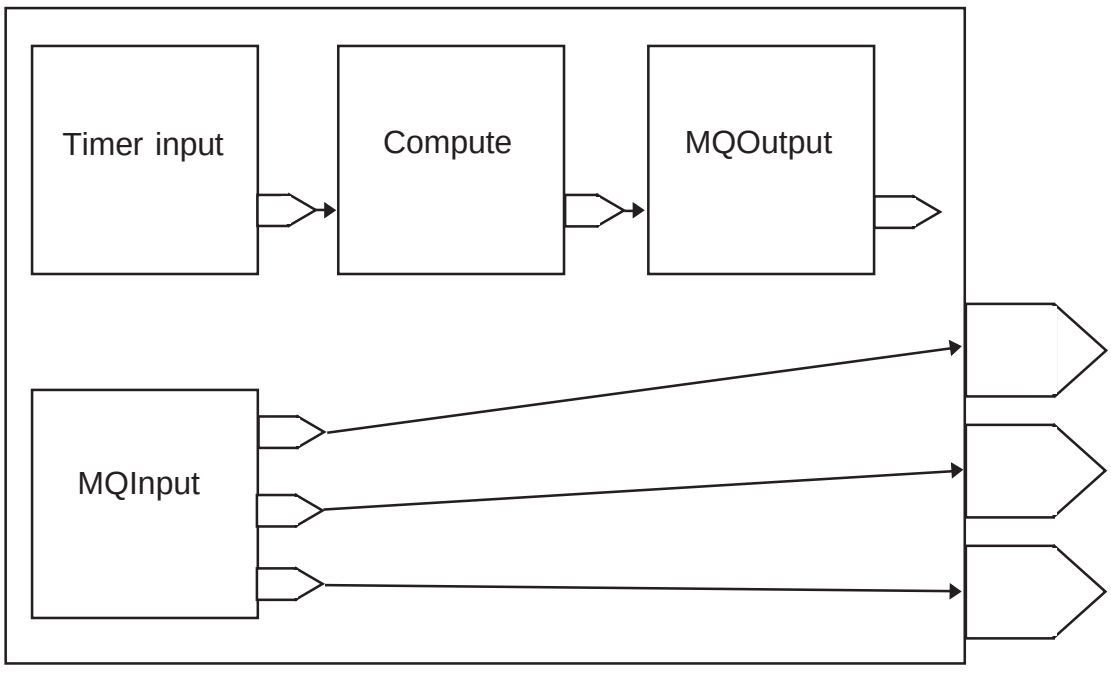

Figure 2: Structure of the self-subscribing input node

needed. This could be simulated by using a very large timeout that results, for example, in one message per month. Since this mechanism was developed also to handle situations where things are not working as expected (for example the subscription is inadvertently deleted), a shorter interval was selected. This way, the subscription is issued once per day. This is not a problem since duplicate subscriptions are ignored by the broker. The overhead for one subscription processing per day was considered to be acceptable.

THE SELF-SUBSCRIBING INPUT NODE

The processing to subscribe a message flow to specific publications is needed for different reasons in WebSphere BI for FN. Therefore, another goal of the development was to create an input node that can be used in any place where a WebSphere BI MB MQInput node is used. This was achieved by creating a sub-flow that is made available as a plug-in for WebSphere BI for FN internal processing. This section describes some alternative implementations.

The sub-flow comprises two different parts, as shown in Figure 2. One part is the WebSphere BI MB MQInput node. The second part is the processing for subscribing the message flow to the defined topics.

As shown in Figure 2, for message flow processing, the MQInput node is the central component of the self-subscribing input node. All output terminals of this node are directly connected as output nodes of the new node. Similarly, all properties of the MQInput node are propagated as properties of the self-subscribing input node. From this perspective, the new node looks identical to an MQInput node.

The process for subscribing the message flow also needs some parameters to perform its tasks. First of all, it needs the information about the input queue name for ‘normal’ processing within the message flow. This information is available to the sub-flow because it’s a mandatory property of the MQInput

node. The information can be accessed by connecting a property from another node to the same property of the self-subscribing input node connected by the input queue name for the MQInput node. This way, the information for the subscription and for the MQInput node are always consistent. The same thing could be done for the topic of the subscription. This would make the topic property of the self-subscribing input node a mandatory property. Whether you implement it this way depends on what you want to do in the 'normal' processing path. If you need the semantics of the topic property of the MQInput node, then you would probably promote another property for the topic of the publication that the message flow needs to process.

WebSphere BI for FN uses the self-subscribing input node for different purposes. In one situation, the message flow needs to process only local broker publications, while in another situation the message flow registers for publications that are issued anywhere in the broker domain. Such options can be specified as part of the subscription request. To allow the specification of such options for the self-subscribing input node, another property can be introduced. This way the node can be used for all kinds of subscriptions.

During the processing of the subscription requests, errors can occur. This could be a permanent failure (like incorrect parameters for the subscription), or a temporary failure (for example, the message flow is not allowed to subscribe to the requested topic). In any case, error handling needs to be carefully defined. You can follow the WebSphere BI MB standard and connect all failure terminals belonging to the nodes needed for subscription processing to the failure terminal of the self-subscribing input node. Another way is for the node itself to send messages to an operator to resolve the problem. Which solution you choose depends on your overall error handling strategy within message flows.

CONCLUSION

Having a self-subscribing input node allows a message flow

to subscribe itself to any kind of publication processed by the WebSphere BI MB message broker. This node can be used in any message flow in the same way as other input nodes, for example instead of an MQInput node. Internally it consists mainly of WebSphere BI MB standard nodes, so the costsof implementing the nodes are minimized.

Michael Groetzner (michael_groetzner@de.ibm.com)
IBM (Germany)

Getting started with configuration event messages – part 2

This month we continue the code to capture configuration event messages.

```
* STRING LIST ATTRIBUTE
 ATTR-STRING-LIST SECTION.
* SET UP ADDRESS OF HEADER AND FIELD
     SET  ADDRESS OF LNK-MQCFSL TO DATA-PTR.
     MOVE DATA-PTR-NUM          TO STRING-PTR-NUM.
     SET  STRING-PTR            TO ADDRESS OF MQCFSL-STRINGLENGTH.
     ADD  4                     TO STRING-PTR-NUM.
     SET  ADDRESS OF LNK-STRING TO STRING-PTR.
* SCAN FOR ATTRIBUTE AND FILL OBJECT ATTRIBUTE TABLE
     PERFORM WITH TEST BEFORE VARYING J FROM 1 BY 1 UNTIL
         J > ASL-NUMBER OR ASL-ID(I, J) = MQCFSL-PARAMETER
     END-PERFORM.
* ATTRIBUTE IS IN PROGRAM TABLE SO ADD IT
     IF ASL-ID(I, J) = MQCFSL-PARAMETER THEN
       SET  ASL-SET(I, J)       TO TRUE
       MOVE MQCFSL-COUNT        TO ASL-COUNT (I, J)
       MOVE MQCFSL-STRINGLENGTH TO ASL-LENGTH (I, J)
* ADD THE NUMBER OF STRINGS CONTAINED IN THE STRING LIST
       IF ASL-COUNT (I, J) = Ø THEN
         MOVE 1 TO ASL-COUNT(I, J)
         MOVE 'NO-ENTRIES' TO ASL-VALUE(I, J, 1)
       ELSE
         PERFORM VARYING K FROM 1 BY 1 UNTIL K > ASL-COUNT (I, J)
           MOVE LNK-STRING (1:MQCFSL-STRINGLENGTH) TO
              ASL-VALUE(I, J, K)
            ADD            MQCFSL-STRINGLENGTH TO STRING-PTR-NUM
```

```
            SET ADDRESS OF LNK-STRING          TO STRING-PTR
          END-PERFORM
        END-IF
      ELSE
* UNKNOWN ATTRIBUTE
        MOVE    MQCFSL-PARAMETER TO W-ERROR-11-NUM
        MOVE    SPACES           TO W-SYSPRINT-DATA
        STRING  '||| UNKNOWN STRINGLIST PARAMETER ' W-ERROR-11
                 DELIMITED BY SIZE INTO W-SYSPRINT-DATA
        PERFORM PRINT-LINE
      END-IF.
* SKIP OVER PROCESSED STRING LIST ENTRIY
      ADD MQCFSL-STRUCLENGTH TO DATA-PTR-NUM.
* AND RETURN
 ATTR-STRING-LIST-END.
      EXIT.
      EJECT
* STRING ATTRIBUTE
 ATTR-STRING SECTION.
* SET UP ADDRESS OF HEADER AND FIELD
      SET  ADDRESS OF LNK-MQCFST TO DATA-PTR.
      MOVE DATA-PTR-NUM          TO STRING-PTR-NUM.
      ADD  LENGTH OF LNK-MQCFST  TO STRING-PTR-NUM.
      SET  ADDRESS OF LNK-STRING TO STRING-PTR.
      EVALUATE MQCFST-PARAMETER
* GET SOME VALUES OUT OF HEADER
        WHEN MQCACF-EVENT-USER-ID
          MOVE LNK-STRING (1:MQCFST-STRINGLENGTH) TO WEH1-USER
        WHEN MQCACF-EVENT-Q-MGR
          MOVE LNK-STRING (1:MQCFST-STRINGLENGTH) TO WEH1-QMGR
* MAYBE A COBOL GURU WILL DO THIS BETTER....
        WHEN MQCACH-CHANNEL-NAME
          IF WEH1-OBJECTTYPE = 'CHANNEL' THEN
            MOVE LNK-STRING (1:MQCFST-STRINGLENGTH) TO
                                                    WEH1-OBJECTNAME
          END-IF
        WHEN  MQCA-NAMELIST-NAME
          IF WEH1-OBJECTTYPE = 'NAMELIST' THEN
            MOVE LNK-STRING (1:MQCFST-STRINGLENGTH) TO
                                                    WEH1-OBJECTNAME
          END-IF
        WHEN MQCA-PROCESS-NAME
          IF WEH1-OBJECTTYPE = 'PROCESS' THEN
            MOVE LNK-STRING (1:MQCFST-STRINGLENGTH) TO
                                                    WEH1-OBJECTNAME
          END-IF
        WHEN MQCA-Q-NAME
          IF WEH1-OBJECTTYPE = 'QUEUE'   THEN
            MOVE LNK-STRING (1:MQCFST-STRINGLENGTH) TO
                                                    WEH1-OBJECTNAME
```

```
            END-IF
          WHEN MQCA-STORAGE-CLASS
            IF WEH1-OBJECTTYPE = 'STGCLASS' THEN
              MOVE LNK-STRING (1:MQCFST-STRINGLENGTH) TO
                                                    WEH1-OBJECTNAME
            END-IF
          WHEN MQCA-Q-MGR-NAME
            IF WEH1-OBJECTTYPE = 'QMGR' THEN
              MOVE LNK-STRING (1:MQCFST-STRINGLENGTH) TO
                                                    WEH1-OBJECTNAME
            END-IF
          WHEN MQCA-AUTH-INFO-NAME
            IF WEH1-OBJECTTYPE = 'AUTHINFO' THEN
              MOVE LNK-STRING (1:MQCFST-STRINGLENGTH) TO
                                                    WEH1-OBJECTNAME
            END-IF
          WHEN MQCA-CF-STRUC-NAME
            IF WEH1-OBJECTTYPE = 'CFSTRUCT' THEN
              MOVE LNK-STRING (1:MQCFST-STRINGLENGTH) TO
                                                    WEH1-OBJECTNAME
            END-IF
        END-EVALUATE.
      * SEARCH FOR ATTRIBUTE IN PROGRAM TABLE
        PERFORM WITH TEST BEFORE VARYING J FROM 1 BY 1 UNTIL
            J > AS-NUMBER OR AS-ID(I, J) = MQCFST-PARAMETER
        END-PERFORM.
      * ATTRIBUTE FOUND, SO PUT INTO TABLE
        IF AS-ID(I, J) = MQCFST-PARAMETER THEN
          MOVE LNK-STRING (1:MQCFST-STRINGLENGTH) TO AS-VALUE(I, J)
          MOVE MQCFST-STRINGLENGTH                TO AS-LENGTH(I, J)
          SET  AS-SET(I, J)                       TO TRUE
      * UNKNOWN STRING ATTRIBUTE
        ELSE
          MOVE    MQCFST-PARAMETER  TO W-ERROR-11-NUM
          MOVE    SPACES            TO W-SYSPRINT-DATA
          STRING  '||| UNKNOWN STRING PARAMETER ' W-ERROR-11
                  DELIMITED BY SIZE INTO W-SYSPRINT-DATA
          PERFORM PRINT-LINE
        END-IF.
      * SKIP OVER PROCESSED ATTRIBUTE
        ADD MQCFST-STRUCLENGTH TO DATA-PTR-NUM.
      * AND RETURN
       ATTR-STRING-END.
        EXIT.
        EJECT
      * STRING INTEGER ATTRIBUTE
       ATTR-INTEGER SECTION.
      * SET UP ADDRESS OF HEADER AND FIELD
        SET ADDRESS OF LNK-MQCFIN TO DATA-PTR.
      * CHECK VARIOUS FIELDS NEEDED FOR HEADER PRINTOUT
```

```
EVALUATE MQCFIN-PARAMETER
  WHEN MQIACF-OBJECT-TYPE
    EVALUATE MQCFIN-VALUE
      WHEN MQOT-CHANNEL
        MOVE 'CHANNEL'   TO WEH1-OBJECTTYPE
      WHEN MQOT-NAMELIST
        MOVE 'NAMELIST'  TO WEH1-OBJECTTYPE
      WHEN MQOT-PROCESS
        MOVE 'PROCESS'   TO WEH1-OBJECTTYPE
      WHEN MQOT-Q
        MOVE 'QUEUE'     TO WEH1-OBJECTTYPE
      WHEN MQOT-STORAGE-CLASS
        MOVE 'STGCLASS'  TO WEH1-OBJECTTYPE
      WHEN MQOT-Q-MGR
        MOVE 'QMGR'      TO WEH1-OBJECTTYPE
      WHEN MQOT-AUTH-INFO
        MOVE 'AUTHINFO'  TO WEH1-OBJECTTYPE
      WHEN MQOT-CF-STRUC
         MOVE 'CFSTRUCT' TO WEH1-OBJECTTYPE
      WHEN OTHER
        MOVE '??UNKNOWN' TO WEH1-OBJECTTYPE
    END-EVALUATE
  WHEN MQIA-QSG-DISP
    EVALUATE MQCFIN-VALUE
      WHEN MQQSGD-Q-MGR
        MOVE 'QMGR'      TO WEH1-DISP
      WHEN MQQSGD-SHARED
        MOVE 'SHARED'    TO WEH1-DISP
      WHEN MQQSGD-GROUP
        MOVE 'GROUP'     TO WEH1-DISP
      WHEN MQQSGD-COPY
        MOVE 'COPY'      TO WEH1-DISP
      WHEN OTHER
        MOVE '??UNKNOWN' TO WEH1-DISP
    END-EVALUATE
  WHEN MQIACF-EVENT-ORIGIN
    EVALUATE MQCFIN-VALUE
      WHEN MQEVO-CONSOLE
        MOVE    'BY CONSOLE ' TO WEH1-HOW
      WHEN MQEVO-INIT
        MOVE    'BY INPUT DATA SET   ' TO WEH1-HOW
      WHEN MQEVO-MSG
        MOVE    'BY SYSTEM.COMMAND.Q ' TO WEH1-HOW
      WHEN MQEVO-MQSET
        MOVE    'BY MQSET CALL       ' TO WEH1-HOW
      WHEN MQEVO-INTERNAL
        MOVE    'BY QMGR INTERNALY   ' TO WEH1-HOW
      WHEN MQEVO-OTHER
        MOVE    'BY OTHERS           ' TO WEH1-HOW
      WHEN OTHER
```

```
            MOVE    '?|? UNKNOWN VALUE   ' TO WEH1-HOW
         END-EVALUATE
     END-EVALUATE.
* CHECK INTERNAL PROGRAM TABLE FOR ATTRIBUTE
     PERFORM WITH TEST BEFORE VARYING J FROM 1 BY 1 UNTIL
         J > AI-NUMBER OR AI-ID(I, J) = MQCFIN-PARAMETER
     END-PERFORM.
* ATTRIBUTE FOUND, SO PUT IT INTO TABLE
     IF AI-ID(I, J) = MQCFIN-PARAMETER THEN
       MOVE    MQCFIN-VALUE TO AI-VALUE (I, J)
       SET     AI-SET(I, J) TO TRUE
       PERFORM EXPLAIN-NUMBER
     ELSE
* ATTRIBUTE NOT FOUND
        MOVE    MQCFIN-PARAMETER  TO W-ERROR-11-NUM
        MOVE    SPACES            TO W-SYSPRINT-DATA
        STRING  '||| UNKNOWN NUMERIC PARAMETER ' W-ERROR-11
                DELIMITED BY SIZE INTO W-SYSPRINT-DATA
        PERFORM PRINT-LINE
     END-IF.
* SKIP OVER PROCESSED ENTRY
     ADD MQCFIN-STRUCLENGTH TO DATA-PTR-NUM.
 GET-INTEGER-ATTRIBUTE-END.
     EXIT.
 ATTR-INTEGER-END.
     EXIT.
     EJECT
* CHECK WHAT ATTRIBUTE DETAIL LEVEL WAS REQUIRED FOR OUTPUT
 OBJECT-DETAILS SECTION.
* CHECK IF DETAILS SHOULD BE PRINTED
     IF (OBJECT-CHANGE)                 OR
        (OBJECT-REFRESH AND REF-DET-ON) OR
        (OBJECT-CREATE  AND CRE-DET-ON) OR
        (OBJECT-DELETE  AND DEL-DET-ON) OR
        (PRINT-ATTRIBUTE-1)             OR
        (PRINT-ATTRIBUTE-2)             THEN
* OKAY, PRINT THE DETAILS
       MOVE    1 TO I
       MOVE    2 TO K
       PERFORM DO-OBJECT-DETAILS-INTEGER VARYING J FROM 1 BY 1
                                          UNTIL J > AI-NUMBER
       PERFORM DO-OBJECT-DETAILS-STRING  VARYING J FROM 1 BY 1
                                          UNTIL J > AS-NUMBER
       PERFORM DO-OBJECT-DETAILS-SLIST   VARYING J FROM 1 BY 1
                                          UNTIL J > ASL-NUMBER
     END-IF.
 OBJECT-DETAILS-END.
     EXIT.
     EJECT
* GO FOR INTEGER OBJECT DETAILS
```

```
 DO-OBJECT-DETAILS-INTEGER SECTION.
* CHECK INTEGER ATTRIBUTE 1, IF NOT SET THEN DO NOT PRINT
     IF (NOT AI-SET(I, J)) OR
* PRINT ONLY INTEGER ATTRIBUTE 2 REQUESTED - SO DON'T PRINT
        (PRINT-ATTRIBUTE-2)  OR
*  OBJECT WAS CHANGED, AND NO DETAILS REQUESTED AND ATTRIBUTES
*  ARE EQUAL) - SO DON'T PRINT
         (OBJECT-CHANGE AND NOT ALT-DET-ON AND
          AI-VALUE(I, J) = AI-VALUE (K, J)) THEN
       NEXT SENTENCE
     ELSE
* ABOVE CONDITIONS DO NOT MATCH, SO PRINT DETAILS FOR
* INTEGER ATTRIBUTE 1
       PERFORM DO-OBJECT-DETAILS-INTEGER1
     END-IF.
* CHECK INTEGER ATTRIBUTE 2, IF NOT SET THEN DO NOT PRINT
     IF (NOT AI-SET(K, J)) OR
* PRINT ONLY INTEGER ATTRIBUTE 1 REQUESTED - SO DON'T PRINT
        (PRINT-ATTRIBUTE-1)  OR
* OBJECT CHANGED BUT ATTRIBUTES MATCH - SO DON'T PRINT
        (OBJECT-CHANGE AND
         AI-VALUE(I, J) = AI-VALUE (K, J)) THEN
       NEXT SENTENCE
     ELSE
* ABOVE CONDITIONS DO NOT MATCH, SO PRINT DETAILS FOR
* INTEGER ATTRIBUTE 2
       PERFORM DO-OBJECT-DETAILS-INTEGER2
     END-IF.
 DO-OBJECT-DETAILS-INTEGER-END.
     EXIT.
* PRINT DETAILS FOR THIS INTEGER ATTRIBUTE 1
 DO-OBJECT-DETAILS-INTEGER1 SECTION.
     MOVE    AI-EXP (I, J)            TO WED1-DESC.
     MOVE    W-EVENT-DETAIL-1 TO W-SYSPRINT-DATA.
     PERFORM PRINT-LINE.
 DO-OBJECT-DETAILS-INTEGER1-END.
     EXIT.
* PRINT DETAILS FOR THIS INTEGER ATTRIBUTE 2
 DO-OBJECT-DETAILS-INTEGER2 SECTION.
     MOVE    AI-EXP (K, J)            TO WED2-DESC.
     MOVE    W-EVENT-DETAIL-2 TO W-SYSPRINT-DATA.
     PERFORM PRINT-LINE.
 DO-OBJECT-DETAILS-INTEGER2-END.
     EXIT.
     EJECT
* GO FOR STRING  OBJECT DETAILS
 DO-OBJECT-DETAILS-STRING SECTION.
* CHECK STRING ATTRIBUTE 1, IF NOT SET THEN DO NOT PRINT
     IF (NOT AS-SET(I, J)) OR
* PRINT ONLY STRING ATTRIBUTE 2 REQUESTED - SO DON NOT PRINT
```

```
        (PRINT-ATTRIBUTE-2)  OR
*  OBJECT WAS CHANGED, AND NO DETAILS REQUESTED AND ATTRIBUTES
*  ARE EQUAL - SO DO NOT PRINT
        (OBJECT-CHANGE AND NOT ALT-DET-ON AND
         AS-VALUE(I, J) = AS-VALUE (K, J)) OR
* DO NOT HANDLE ALTERATION AND CREATION DATE/TIME AS
* CHANGED ATTRIBUTES
        (OBJECT-CHANGE AND NOT ALT-DET-ON AND (
        (AS-ID(I, J) = MQCA-ALTERATION-DATE OR
         AS-ID(I, J) = MQCA-ALTERATION-TIME OR
         AS-ID(I, J) = MQCA-CREATION-DATE OR
         AS-ID(I, J) = MQCA-CREATION-TIME))) THEN
       NEXT SENTENCE
     ELSE
* ABOVE CONDITIONS DO NOT MATCH, SO PRINT DETAILS FOR
* STRING ATTRIBUTE 1
       PERFORM DO-OBJECT-DETAILS-STRING1
     END-IF.
* CHECK STRING ATTRIBUTE 2 - IF NOT SET THEN DO NOT PRINT
     IF (NOT AS-SET(K, J)) OR
* PRINT ONLY STRING ATTRIBUTE 1 REQUESTED - SO DO NOT PRINT
        (PRINT-ATTRIBUTE-1)  OR
* OBJECT CHANGED BUT ATTRIBUTES MATCH - SO DO NOT PRINT
        (OBJECT-CHANGE AND
         AS-VALUE(I, J) = AS-VALUE (K, J)) OR
* DO NOT HANDLE ALTERATION AND CREATION DATE/TIME AS
* CHANGED ATTRIBUTES
        (OBJECT-CHANGE AND NOT ALT-DET-ON AND (
        (AS-ID(K, J) = MQCA-ALTERATION-DATE OR
         AS-ID(K, J) = MQCA-ALTERATION-TIME OR
         AS-ID(K, J) = MQCA-CREATION-DATE OR
         AS-ID(K, J) = MQCA-CREATION-TIME))) THEN
       NEXT SENTENCE
     ELSE
* ABOVE CONDITIONS DO NOT MATCH, SO PRINT DETAILS FOR
* STRING ATTRIBUTE 2
       PERFORM DO-OBJECT-DETAILS-STRING2
     END-IF.
 DO-OBJECT-DETAILS-STRING-END.
     EXIT.
* PRINT DETAILS FOR STRING ATTRIBUTE 1
 DO-OBJECT-DETAILS-STRING1 SECTION.
     MOVE    SPACES         TO WED1-DESC.
* FIND OUT ABOUT END OF STRING
     MOVE    AS-VALUE(I, J) TO STRING256.
     PERFORM FIND-END-OF-STRING.
     STRING  AS-DESC  (I, J) '('   DELIMITED BY SPACE
             AS-VALUE (I, J)(1:S2) DELIMITED BY SIZE
             ')'                   DELIMITED BY SPACE
             INTO WED1-DESC.
```

```
     MOVE    W-EVENT-DETAIL-1 TO W-SYSPRINT-DATA.
     PERFORM PRINT-LINE.
 DO-OBJECT-DETAILS-STRING1-END.
     EXIT.
* PRINT DETAILS FOR STRING ATTRIBUTE 2
 DO-OBJECT-DETAILS-STRING2 SECTION.
     MOVE    SPACES         TO WED2-DESC.
* FIND OUT ABOUT END OF STRING
     MOVE    AS-VALUE(K, J) TO STRING256.
     PERFORM FIND-END-OF-STRING.
     STRING  AS-DESC (K, J) '('    DELIMITED BY SPACE
             AS-VALUE (K, J)(1:S2) DELIMITED BY SIZE
             ')'                   DELIMITED BY SPACE
             INTO WED2-DESC.
     MOVE    W-EVENT-DETAIL-2 TO W-SYSPRINT-DATA.
     PERFORM PRINT-LINE.
 DO-OBJECT-DETAILS-STRING2-END.
     EXIT.
     EJECT
* GO FOR STRINGLIST OBJECT DETAILS
 DO-OBJECT-DETAILS-SLIST  SECTION.
* CHECK STRINGLIST ATTRIBUTE 1 - IF NOT SET THEN DO NOT PRINT
     IF (NOT ASL-SET(I, J)) OR
* PRINT ONLY ATTRIBUTE 2 REQUESTED - SO DON NOT PRINT
        (PRINT-ATTRIBUTE-2)  OR
*  OBJECT WAS CHANGED, AND NO DETAILS REQUESTED AND ATTRIBUTES
*  ARE EQUAL - SO DO NOT PRINT
        (OBJECT-CHANGE AND NOT ALT-DET-ON AND
         ASL-VALUE1(I, J) = ASL-VALUE1 (K, J)) THEN
       NEXT SENTENCE
     ELSE
* ABOVE CONDITIONS DO NOT MATCH, SO PRINT DETAILS FOR
* STRINGLIST ATTRIBUTE 1
       PERFORM DO-OBJECT-DETAILS-SLIST1
     END-IF.
* CHECK STRINGLIST ATTRIBUTE 2 - IF NOT SET THEN DO NOT PRINT
     IF (NOT ASL-SET(K, J)) OR
* PRINT ONLY STRINGLIST ATTRIBUTE 1 REQUESTED - SO DO NOT PRINT
        (PRINT-ATTRIBUTE-1)  OR
* OBJECT CHANGED BUT ATTRIBUTES MATCH - SO DO NOT PRINT
        (OBJECT-CHANGE AND
         ASL-VALUE1(I, J) = ASL-VALUE1 (K, J)) THEN
       NEXT SENTENCE
     ELSE
* ABOVE CONDITIONS DO NOT MATCH, SO PRINT DETAILS FOR
* STRINGLIST ATTRIBUTE 2
       PERFORM DO-OBJECT-DETAILS-SLIST2
     END-IF.
 DO-OBJECT-DETAILS-SLIST-END.
     EXIT.
```

```
     EJECT
* PRINT STRINGLIST ATTRIBUTE 1
* IF STRINGLIST ELEMENTS CONTAIN BLANKS THEY WILL BE TRUNCATED
* BY THE "STRING... DELIMITED BY SPACE" COMMAND.
* IF SO, THE "FIND-END-OF-STRING" USED FOR STRINGS HAS TO BE
* USED HERE (CURRENTLY WE USE JUST THE STRING COMMAND)
 DO-OBJECT-DETAILS-SLIST1 SECTION.
     MOVE SPACES TO WED1-DESC.
     IF ASL-COUNT (I, J) = 1 THEN
       STRING  ASL-DESC (I, J) '(' ASL-VALUE (I, J, 1) ')'
                    DELIMITED BY SPACE INTO WED1-DESC
       MOVE    W-EVENT-DETAIL-1 TO           W-SYSPRINT-DATA
       PERFORM PRINT-LINE
     ELSE
* FIRST ITEM IN LIST
       STRING  ASL-DESC (I, J) '(' ASL-VALUE (I, J, 1) ','
                    DELIMITED BY SPACE INTO WED1-DESC
       MOVE    W-EVENT-DETAIL-1 TO           W-SYSPRINT-DATA
       PERFORM PRINT-LINE
* SECOND ITEM TO N-1 ITEM
       PERFORM VARYING L FROM 2 BY 1 UNTIL L = ASL-COUNT(I, J)
         MOVE    SPACES                   TO   WED1-DESC
         STRING  ASL-VALUE (I, J, L) ','
                      DELIMITED BY SPACE INTO WED1-DESC
         MOVE    W-EVENT-DETAIL-1 TO           W-SYSPRINT-DATA
         PERFORM PRINT-LINE
       END-PERFORM
* LAST ITEM IN LIST
       MOVE    SPACES                   TO   WED1-DESC
       STRING  ASL-VALUE (I, J, L) ')'
                    DELIMITED BY SPACE INTO WED1-DESC
       MOVE    W-EVENT-DETAIL-1 TO           W-SYSPRINT-DATA
       PERFORM PRINT-LINE
     END-IF.
 DO-OBJECT-DETAILS-SLIST1-END.
     EXIT.
* PRINT STRINGLIST ATTRIBUTE 2
 DO-OBJECT-DETAILS-SLIST2 SECTION.
     MOVE SPACES TO WED2-DESC.
     IF ASL-COUNT (K, J) = 1 THEN
       STRING  ASL-DESC (K, J) '(' ASL-VALUE (K, J, 1) ')'
                    DELIMITED BY SPACE INTO WED2-DESC
       MOVE    W-EVENT-DETAIL-2 TO           W-SYSPRINT-DATA
       PERFORM PRINT-LINE
     ELSE
* FIRST ITEM IN LIST
       STRING  ASL-DESC (K, J) '(' ASL-VALUE (K, J, 1) ','
                    DELIMITED BY SPACE INTO WED2-DESC
       MOVE    W-EVENT-DETAIL-2 TO           W-SYSPRINT-DATA
       PERFORM PRINT-LINE
```

```
* SECOND ITEM TO N-1 ITEM
       PERFORM VARYING L FROM 2 BY 1 UNTIL L = ASL-COUNT(K, J)
         MOVE    SPACES                    TO   WED2-DESC
         STRING  ASL-VALUE (K, J, L) ','
                       DELIMITED BY SPACE INTO WED2-DESC
         MOVE    W-EVENT-DETAIL-2 TO           W-SYSPRINT-DATA
         PERFORM PRINT-LINE
       END-PERFORM
* LAST ITEM IN LIST
       MOVE   SPACES                     TO   WED2-DESC
       STRING ASL-VALUE (K, J, L) ')'
                     DELIMITED BY SPACE INTO WED2-DESC
       MOVE    W-EVENT-DETAIL-2 TO           W-SYSPRINT-DATA
       PERFORM PRINT-LINE
     END-IF.
 DO-OBJECT-DETAILS-SLIST2-END.
     EXIT.
     EJECT
* GET PROPER CHARACTER DESCRIPTION FOR INTEGER ATTRIBUTE
 EXPLAIN-NUMBER SECTION.
     MOVE SPACES TO AI-EXP (I, J).
     EVALUATE MQCFIN-PARAMETER
       WHEN MQIA-AUTH-INFO-TYPE
* MQIAT-CRL-LDAP DOES NOT EXIST IN COPY BOOKS
*        EVALUATE MQCFIN-VALUE
*          WHEN MQIAT-CRL-LDAP
             MOVE 'AUTHTYPE(CRLLDAP)' TO AI-EXP (I, J)
*          WHEN OTHER
*            MOVE '|| AUTHTYPE(UNKNOWN)' TO AI-EXP (I, J)
*        END-EVALUATE
       WHEN MQIA-CF-LEVEL
         PERFORM EXP-NUMBER
       WHEN MQIA-CF-RECOVER
          PERFORM YES-NO
       WHEN MQIACH-CHANNEL-TYPE
         EVALUATE MQCFIN-VALUE
           WHEN MQCHT-SENDER
             MOVE 'SDR'      TO WRK-OTHER
           WHEN MQCHT-SERVER
             MOVE 'SVR'      TO WRK-OTHER
           WHEN MQCHT-RECEIVER
             MOVE 'RCVR'     TO WRK-OTHER
           WHEN MQCHT-REQUESTER
             MOVE 'RQSTR'    TO WRK-OTHER
           WHEN MQCHT-SVRCONN
             MOVE 'SVRCONN'  TO WRK-OTHER
           WHEN MQCHT-CLNTCONN
             MOVE 'CLNTCONN' TO WRK-OTHER
           WHEN MQCHT-CLUSRCVR
             MOVE 'CLUSRCVR' TO WRK-OTHER
```

```
    WHEN MQCHT-CLUSSDR
      MOVE 'CLUSSDR'  TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?'      TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIACH-XMIT-PROTOCOL-TYPE
  EVALUATE MQCFIN-VALUE
    WHEN MQXPT-LU62
      MOVE 'LU62'    TO WRK-OTHER
    WHEN MQXPT-TCP
      MOVE 'TCP'     TO WRK-OTHER
    WHEN MQXPT-NETBIOS
      MOVE 'NETBIOS' TO WRK-OTHER
    WHEN MQXPT-SPX
      MOVE 'SPX'     TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?'     TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIACH-BATCH-SIZE
  PERFORM EXP-NUMBER
WHEN MQIACH-DISC-INTERVAL
  PERFORM EXP-NUMBER
WHEN MQIACH-SHORT-RETRY
  PERFORM EXP-NUMBER
WHEN MQIACH-SHORT-TIMER
  PERFORM EXP-NUMBER
WHEN MQIACH-LONG-RETRY
  PERFORM EXP-NUMBER
WHEN MQIACH-LONG-TIMER
  PERFORM EXP-NUMBER
WHEN MQIACH-DATA-CONVERSION
  EVALUATE MQCFIN-VALUE
    WHEN MQCDC-NO-SENDER-CONVERSION
      MOVE 'NO'  TO WRK-OTHER
    WHEN MQCDC-SENDER-CONVERSION
      MOVE 'YES' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIACH-PUT-AUTHORITY
  EVALUATE MQCFIN-VALUE
    WHEN MQPA-DEFAULT
      MOVE 'DEF                ' TO WRK-OTHER
    WHEN MQPA-CONTEXT
      MOVE 'CTX                ' TO WRK-OTHER
    WHEN MQPA-ALTERNATE-OR-MCA
      MOVE 'MCA                ' TO WRK-OTHER
```

```
    WHEN MQPA-ONLY-MCA
      MOVE 'ONLYMCA             ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIACH-SEQUENCE-NUMBER-WRAP
  PERFORM EXP-NUMBER
WHEN MQIACH-MAX-MSG-LENGTH
  PERFORM EXP-NUMBER
WHEN MQIACH-MCA-TYPE
  EVALUATE MQCFIN-VALUE
    WHEN MQMCAT-PROCESS
      MOVE 'PROCESS             ' TO WRK-OTHER
    WHEN MQMCAT-THREAD
      MOVE 'THREAD              ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIACH-BATCH-INTERVAL
  PERFORM EXP-NUMBER
WHEN MQIACH-HB-INTERVAL
  PERFORM EXP-NUMBER
WHEN MQIACH-NPM-SPEED
  EVALUATE MQCFIN-VALUE
    WHEN MQNPMS-NORMAL
      MOVE 'NORMAL              ' TO WRK-OTHER
    WHEN MQNPMS-FAST
      MOVE 'FAST                ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIACH-NETWORK-PRIORITY
  PERFORM EXP-NUMBER
WHEN MQIACH-BATCH-HB
  PERFORM EXP-NUMBER
WHEN MQIACH-KEEP-ALIVE-INTERVAL
  PERFORM EXP-NUMBER
WHEN MQIACH-SSL-CLIENT-AUTH
  EVALUATE MQCFIN-VALUE
    WHEN MQSCA-REQUIRED
      MOVE 'REQUIRED            ' TO WRK-OTHER
    WHEN MQSCA-OPTIONAL
      MOVE 'OPTIONAL            ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
```

```
       WHEN MQIA-NAMELIST-TYPE
* VALUES NOT SPECIFIED IN COPY BOOK, HAD TO FIND OUT MYSELF
         EVALUATE MQCFIN-VALUE
           WHEN Ø
             MOVE 'NONE                ' TO WRK-OTHER
           WHEN 1
             MOVE 'QUEUE               ' TO WRK-OTHER
           WHEN 2
             MOVE 'CLUSTER             ' TO WRK-OTHER
           WHEN 3
             MOVE 'AUTHINFO            ' TO WRK-OTHER
           WHEN OTHER
             MOVE '?|?' TO WRK-OTHER
         END-EVALUATE
         PERFORM STRING-OTHER
       WHEN MQIA-NAME-COUNT
         PERFORM EXP-NUMBER
       WHEN MQIA-APPL-TYPE
         EVALUATE MQCFIN-VALUE
           WHEN MQAT-CICS
             MOVE 'CICS                ' TO WRK-OTHER
           WHEN MQAT-DEFAULT
             MOVE 'DEF                 ' TO WRK-OTHER
           WHEN MQAT-DOS
             MOVE 'DOS                 ' TO WRK-OTHER
           WHEN MQAT-IMS
             MOVE 'IMS                 ' TO WRK-OTHER
           WHEN MQAT-MVS
             MOVE 'MVS                 ' TO WRK-OTHER
           WHEN MQAT-NOTES-AGENT
             MOVE 'NOTESAGENT          ' TO WRK-OTHER
           WHEN MQAT-NSK
             MOVE 'NSK                 ' TO WRK-OTHER
           WHEN MQAT-VMS
             MOVE 'VMS                 ' TO WRK-OTHER
           WHEN MQAT-OS2
             MOVE 'OS2                 ' TO WRK-OTHER
           WHEN MQAT-OS4ØØ
             MOVE 'OS499               ' TO WRK-OTHER
           WHEN MQAT-UNIX
             MOVE 'UNIX                ' TO WRK-OTHER
           WHEN MQAT-WINDOWS
             MOVE 'WINDOWS             ' TO WRK-OTHER
           WHEN MQAT-WINDOWS-NT
             MOVE 'WINDOWSNT           ' TO WRK-OTHER
* THERE ARE STILL OTHER MQAT-* IN SCSQCOBC
           WHEN OTHER
             MOVE '?|?' TO WRK-OTHER
         END-EVALUATE
         PERFORM STRING-OTHER
```

```
WHEN MQIA-Q-TYPE
  EVALUATE MQCFIN-VALUE
    WHEN MQQT-ALIAS
      MOVE 'QALIAS              ' TO WRK-OTHER
    WHEN MQQT-LOCAL
      MOVE 'QLOCAL              ' TO WRK-OTHER
    WHEN MQQT-REMOTE
      MOVE 'QREMOTE             ' TO WRK-OTHER
    WHEN MQQT-MODEL
      MOVE 'QMODEL              ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIA-INHIBIT-GET
  EVALUATE MQCFIN-VALUE
    WHEN MQQA-GET-ALLOWED
      MOVE 'ENABLED             ' TO WRK-OTHER
    WHEN MQQA-GET-INHIBITED
      MOVE 'DISABLED            ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIA-INHIBIT-PUT
  EVALUATE MQCFIN-VALUE
    WHEN MQQA-PUT-ALLOWED
      MOVE 'ENABLED             ' TO WRK-OTHER
    WHEN MQQA-PUT-INHIBITED
      MOVE 'DISABLED            ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIA-DEF-PRIORITY
  PERFORM EXP-NUMBER
WHEN MQIA-DEF-PERSISTENCE
  EVALUATE MQCFIN-VALUE
    WHEN MQPER-PERSISTENT
      MOVE 'YES                 ' TO WRK-OTHER
    WHEN MQPER-NOT-PERSISTENT
      MOVE 'NO                  ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIA-MAX-Q-DEPTH
  PERFORM EXP-NUMBER
WHEN MQIA-MAX-MSG-LENGTH
  PERFORM EXP-NUMBER
```

```
WHEN MQIA-BACKOUT-THRESHOLD
  PERFORM EXP-NUMBER
WHEN MQIA-SHAREABILITY
  EVALUATE MQCFIN-VALUE
    WHEN MQQA-SHAREABLE
      MOVE 'YES                ' TO WRK-OTHER
    WHEN MQQA-NOT-SHAREABLE
      MOVE 'NO                 ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIA-DEF-INPUT-OPEN-OPTION
  EVALUATE MQCFIN-VALUE
    WHEN MQOO-INPUT-EXCLUSIVE
      MOVE 'EXCL               ' TO WRK-OTHER
    WHEN MQOO-INPUT-SHARED
      MOVE 'SHARED             ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIA-HARDEN-GET-BACKOUT
  EVALUATE MQCFIN-VALUE
    WHEN MQQA-BACKOUT-HARDENED
      MOVE 'HARDENBO           ' TO AI-EXP (I, J)
    WHEN MQQA-BACKOUT-NOT-HARDENED
      MOVE 'NOHARDENBO         ' TO AI-EXP (I, J)
    WHEN OTHER
      MOVE 'HARDENBO(?|?|?)    ' TO AI-EXP (I, J)
  END-EVALUATE
WHEN MQIA-MSG-DELIVERY-SEQUENCE
  EVALUATE MQCFIN-VALUE
    WHEN MQMDS-PRIORITY
      MOVE 'PRIORITY           ' TO WRK-OTHER
    WHEN MQMDS-FIFO
      MOVE 'FIFO               ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIA-RETENTION-INTERVAL
  PERFORM EXP-NUMBER
WHEN MQIA-DEFINITION-TYPE
  EVALUATE MQCFIN-VALUE
    WHEN MQQDT-PREDEFINED
      MOVE 'PREDEFINED         ' TO WRK-OTHER
    WHEN MQQDT-PERMANENT-DYNAMIC
      MOVE 'PERMANANT DYNAMIC  ' TO WRK-OTHER
    WHEN MQQDT-SHARED-DYNAMIC
```

```
            MOVE 'SHARED DYNAMIC      ' TO WRK-OTHER
          WHEN MQQDT-PREDEFINED
            MOVE 'PREDEFINED          ' TO WRK-OTHER
          WHEN OTHER
            MOVE '?|?' TO WRK-OTHER
        END-EVALUATE
        PERFORM STRING-OTHER
      WHEN MQIA-USAGE
        EVALUATE MQCFIN-VALUE
          WHEN MQUS-NORMAL
            MOVE 'NORMAL              ' TO WRK-OTHER
          WHEN MQUS-TRANSMISSION
            MOVE 'XMITQ               ' TO WRK-OTHER
          WHEN MQQDT-SHARED-DYNAMIC
            MOVE 'SHARED DYNAMIC      ' TO WRK-OTHER
          WHEN MQQDT-PREDEFINED
            MOVE 'PREDEFINED          ' TO WRK-OTHER
          WHEN OTHER
            MOVE '?|?' TO WRK-OTHER
        END-EVALUATE
        PERFORM STRING-OTHER
      WHEN MQIA-TRIGGER-CONTROL
        EVALUATE MQCFIN-VALUE
          WHEN MQTC-OFF
            MOVE 'NOTRIGGER           ' TO AI-EXP (I, J)
          WHEN MQTC-ON
            MOVE 'TRIGGER             ' TO AI-EXP (I, J)
          WHEN OTHER
            MOVE 'TRIGGER(?|?|?|?)    ' TO AI-EXP (I, J)
        END-EVALUATE
      WHEN MQIA-TRIGGER-TYPE
        EVALUATE MQCFIN-VALUE
          WHEN MQTT-NONE
            MOVE 'NONE                ' TO WRK-OTHER
          WHEN MQTT-FIRST
            MOVE 'FIRST               ' TO WRK-OTHER
          WHEN MQTT-EVERY
            MOVE 'EVERY               ' TO WRK-OTHER
          WHEN MQTT-DEPTH
            MOVE 'DEPTH               ' TO WRK-OTHER
          WHEN OTHER
            MOVE '?|?' TO WRK-OTHER
        END-EVALUATE
        PERFORM STRING-OTHER
      WHEN MQIA-TRIGGER-MSG-PRIORITY
        PERFORM EXP-NUMBER
      WHEN MQIA-TRIGGER-DEPTH
        PERFORM EXP-NUMBER
      WHEN MQIA-Q-DEPTH-HIGH-LIMIT
        PERFORM EXP-NUMBER
```

```
WHEN MQIA-Q-DEPTH-LOW-LIMIT
  PERFORM EXP-NUMBER
WHEN MQIA-Q-SERVICE-INTERVAL
  PERFORM EXP-NUMBER
WHEN MQIA-DEF-BIND
  EVALUATE MQCFIN-VALUE
    WHEN MQBND-BIND-ON-OPEN
      MOVE 'OPEN                ' TO WRK-OTHER
    WHEN MQBND-BIND-NOT-FIXED
      MOVE 'NOTFIXED            ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIA-INDEX-TYPE
  EVALUATE MQCFIN-VALUE
    WHEN MQIT-NONE
      MOVE 'NONE                ' TO WRK-OTHER
    WHEN MQIT-MSG-ID
      MOVE 'MSGID               ' TO WRK-OTHER
    WHEN MQIT-CORREL-ID
      MOVE 'CORRELID            ' TO WRK-OTHER
    WHEN MQIT-MSG-TOKEN
      MOVE 'GROUPID             ' TO WRK-OTHER
    WHEN MQIT-GROUP-ID
      MOVE 'MSGTOKEN            ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIA-PLATFORM
  PERFORM EXP-NUMBER
WHEN MQIA-COMMAND-LEVEL
  PERFORM EXP-NUMBER
WHEN MQIA-CPI-LEVEL
  PERFORM EXP-NUMBER
WHEN MQIA-IGQ-PUT-AUTHORITY
  EVALUATE MQCFIN-VALUE
    WHEN MQIGQPA-DEFAULT
      MOVE 'DEF                 ' TO WRK-OTHER
    WHEN MQIGQPA-CONTEXT
      MOVE 'CTX                 ' TO WRK-OTHER
    WHEN MQIGQPA-ONLY-IGQ
      MOVE 'ONLYIGQ             ' TO WRK-OTHER
    WHEN MQIGQPA-ALTERNATE-OR-IGQ
      MOVE 'ALTIGQ              ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
```

```
WHEN MQIA-INTRA-GROUP-QUEUING
  EVALUATE MQCFIN-VALUE
    WHEN YES-VALUE
      MOVE 'ENABLED             ' TO WRK-OTHER
    WHEN NO-VALUE
      MOVE 'DISABLED            ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIA-EXPIRY-INTERVAL
  PERFORM EXP-NUMBER
WHEN MQIA-SSL-TASKS
  PERFORM EXP-NUMBER
WHEN MQIA-CONFIGURATION-EVENT
  EVALUATE MQCFIN-VALUE
    WHEN MQEVR-ENABLED
      MOVE 'ENABLED             ' TO WRK-OTHER
    WHEN MQEVR-DISABLED
      MOVE 'DISABLED            ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIA-TRIGGER-INTERVAL
  PERFORM EXP-NUMBER
WHEN MQIA-MAX-PRIORITY
  PERFORM EXP-NUMBER
WHEN MQIA-CODED-CHAR-SET-ID
  PERFORM EXP-NUMBER
WHEN MQIA-MAX-HANDLES
  PERFORM EXP-NUMBER
WHEN MQIA-MAX-UNCOMMITTED-MSGS
  PERFORM EXP-NUMBER
WHEN MQIA-SYNCPOINT
  PERFORM EXP-NUMBER
WHEN MQIA-AUTHORITY-EVENT
  EVALUATE MQCFIN-VALUE
    WHEN MQEVR-ENABLED
      MOVE 'ENABLED             ' TO WRK-OTHER
    WHEN MQEVR-DISABLED
      MOVE 'DISABLED            ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIA-INHIBIT-EVENT
  EVALUATE MQCFIN-VALUE
    WHEN MQEVR-ENABLED
      MOVE 'ENABLED             ' TO WRK-OTHER
```

```
    WHEN MQEVR-DISABLED
      MOVE 'DISABLED            ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIA-LOCAL-EVENT
  EVALUATE MQCFIN-VALUE
    WHEN MQEVR-ENABLED
      MOVE 'ENABLED             ' TO WRK-OTHER
    WHEN MQEVR-DISABLED
      MOVE 'DISABLED            ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIA-REMOTE-EVENT
  EVALUATE MQCFIN-VALUE
    WHEN MQEVR-ENABLED
      MOVE 'ENABLED             ' TO WRK-OTHER
    WHEN MQEVR-DISABLED
      MOVE 'DISABLED            ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIA-START-STOP-EVENT
  EVALUATE MQCFIN-VALUE
    WHEN MQEVR-ENABLED
      MOVE 'ENABLED             ' TO WRK-OTHER
    WHEN MQEVR-DISABLED
      MOVE 'DISABLED            ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIA-PERFORMANCE-EVENT
  EVALUATE MQCFIN-VALUE
    WHEN MQEVR-ENABLED
      MOVE 'ENABLED             ' TO WRK-OTHER
    WHEN MQEVR-DISABLED
      MOVE 'DISABLED            ' TO WRK-OTHER
    WHEN OTHER
      MOVE '?|?' TO WRK-OTHER
  END-EVALUATE
  PERFORM STRING-OTHER
WHEN MQIA-CLUSTER-WORKLOAD-LENGTH
  PERFORM EXP-NUMBER
WHEN MQIA-PAGESET-ID
  PERFORM EXP-NUMBER
```

```
        WHEN MQIACF-EVENT-ORIGIN
          EVALUATE MQCFIN-VALUE
            WHEN MQEVO-OTHER
              MOVE 'OTHER               ' TO WRK-OTHER
            WHEN MQEVO-CONSOLE
              MOVE 'CONSOLE             ' TO WRK-OTHER
            WHEN MQEVO-INIT
              MOVE 'INIT                ' TO WRK-OTHER
            WHEN MQEVO-MSG
              MOVE 'MESSAGE             ' TO WRK-OTHER
            WHEN MQEVO-MQSET
              MOVE 'MQSET               ' TO WRK-OTHER
            WHEN MQEVO-INTERNAL
              MOVE 'INTERNAL            ' TO WRK-OTHER
            WHEN OTHER
              MOVE '?|?' TO WRK-OTHER
          END-EVALUATE
          PERFORM STRING-OTHER
        WHEN MQIACF-EVENT-APPL-TYPE
          EVALUATE MQCFIN-VALUE
*     ZOS MVS OS390 ARE THE SAME
            WHEN MQAT-ZOS
              MOVE 'Z/OS                ' TO WRK-OTHER
            WHEN MQAT-XCF
              MOVE 'XCF                 ' TO WRK-OTHER
            WHEN MQAT-CICS-BRIDGE
              MOVE 'CICSBRIDGE          ' TO WRK-OTHER
            WHEN MQAT-NOTES-AGENT
              MOVE 'NOTESAGENT          ' TO WRK-OTHER
            WHEN MQAT-USER
              MOVE 'USER                ' TO WRK-OTHER
            WHEN MQAT-BROKER
              MOVE 'BROKER              ' TO WRK-OTHER
            WHEN MQAT-JAVA
              MOVE 'JAVA                ' TO WRK-OTHER
            WHEN MQAT-DQM
              MOVE 'DQM                 ' TO WRK-OTHER
* THESE ARE NOT VALID FOR CONFIGURATION EVENTS
*           WHEN MQIA-AUTHORITY-EVENT
*             MOVE 'AUTHORITY           ' TO WRK-OTHER
*           WHEN MQIA-CHANNEL-AUTO-DEF-EVENT
*             MOVE 'AUTODEFINITION      ' TO WRK-OTHER
*           WHEN MQIA-CONFIGURATION-EVENT
*             MOVE 'CONFIGURAITON       ' TO WRK-OTHER
*           WHEN MQIA-INHIBIT-EVENT
*             MOVE 'INHIBIT             ' TO WRK-OTHER
*           WHEN MQIA-LOCAL-EVENT
*             MOVE 'LOCAL               ' TO WRK-OTHER
*           WHEN MQIA-PERFORMANCE-EVENT
*             MOVE 'PERFORMANCE         ' TO WRK-OTHER
```

```
*          WHEN MQIA-Q-DEPTH-HIGH-EVENT
*            MOVE 'QUEUEDEPTHHIGH      ' TO WRK-OTHER
*          WHEN MQIA-Q-DEPTH-LOW-EVENT
*            MOVE 'QUEUEDEPTHLOW       ' TO WRK-OTHER
*          WHEN MQIA-Q-DEPTH-MAX-EVENT
*            MOVE 'QUEUEDEPTHMAX       ' TO WRK-OTHER
*          WHEN MQIA-Q-SERVICE-INTERVAL-EVENT
*            MOVE 'QUEUESERVICE        ' TO WRK-OTHER
*          WHEN MQIA-REMOTE-EVENT
*            MOVE 'REMOTE              ' TO WRK-OTHER
*          WHEN MQIA-START-STOP-EVENT
*            MOVE 'START/STOP          ' TO WRK-OTHER
           WHEN OTHER
             MOVE '?|?' TO WRK-OTHER
         END-EVALUATE
         PERFORM STRING-OTHER
       WHEN MQIACF-OBJECT-TYPE
          EVALUATE MQCFIN-VALUE
             WHEN MQOT-CHANNEL
                MOVE 'CHANNEL' TO WRK-OTHER
             WHEN MQOT-NAMELIST
                MOVE 'NAMELIST' TO WRK-OTHER
             WHEN MQOT-PROCESS
                MOVE 'PROCESS' TO WRK-OTHER
             WHEN MQOT-Q
                MOVE 'QUEUE' TO WRK-OTHER
             WHEN MQOT-STORAGE-CLASS
                MOVE 'STGCLASS' TO WRK-OTHER
             WHEN MQOT-Q-MGR
                MOVE 'QMGR'    TO WRK-OTHER
             WHEN MQOT-AUTH-INFO
                MOVE 'AUTHINFO' TO WRK-OTHER
             WHEN MQOT-CF-STRUC
                MOVE 'CFSTRUCT' TO WRK-OTHER
             WHEN OTHER
                MOVE '??UNKNOWN' TO WRK-OTHER
          END-EVALUATE
         PERFORM STRING-OTHER
       WHEN MQIA-QSG-DISP
         EVALUATE MQCFIN-VALUE
           WHEN MQQSGD-Q-MGR
             MOVE 'QMGR                ' TO WRK-OTHER
           WHEN MQQSGD-COPY
             MOVE 'COPY                ' TO WRK-OTHER
           WHEN MQQSGD-SHARED
             MOVE 'SHARED              ' TO WRK-OTHER
           WHEN MQQSGD-GROUP
             MOVE 'GROUP               ' TO WRK-OTHER
           WHEN OTHER
             MOVE '?|?' TO WRK-OTHER
```

```
         END-EVALUATE
         PERFORM STRING-OTHER
* FINALLY, THIS HAS COME TO AN END
     END-EVALUATE.
 EXPLAIN-NUMBER-END.
     EXIT.
* HANDLE NUMBERS IN OUTPUT
 EXP-NUMBER SECTION.
     MOVE MQCFIN-VALUE TO WRK-NUMBER-RIGHT.
     MOVE ZERO TO L.
     MOVE SPACES TO WRK-NUMBER.
     PERFORM VARYING K FROM 1 BY 1 UNTIL K > 13
       IF WRK-NUMBER-ZWO  (K:1) NOT = SPACE THEN
          ADD 1 TO L
          MOVE WRK-NUMBER-ZWO (K:1) TO WRK-NUMBER-X (L)
       END-IF
     END-PERFORM.
     STRING AI-DESC (I, J) '(' WRK-NUMBER ')' DELIMITED BY
            SPACE INTO AI-EXP(I, J).
 EXP-NUMBER-END.
     EXIT.
* HANDLE YES/NO IN OUTPUT
 YES-NO SECTION.
     EVALUATE MQCFIN-VALUE
       WHEN YES-VALUE
         STRING AI-DESC (I, J) '(YES)'
         DELIMITED BY SPACE          INTO AI-EXP (I, J)
       WHEN NO-VALUE
         STRING AI-DESC (I, J) '(NO)'
         DELIMITED BY SPACE          INTO AI-EXP (I, J)
       WHEN OTHER
         STRING '||' AI-DESC (I, J) '(?|?|?)'
         DELIMITED BY SPACE          INTO AI-EXP (I, J)
     END-EVALUATE.
 YES-NO-END.
     EXIT.
* HANDLE STRINGS IN OUTPUT
 STRING-OTHER SECTION.
     STRING AI-DESC (I, J) '(' WRK-OTHER ')'
     DELIMITED BY SPACE          INTO AI-EXP (I, J).
 STRING-OTHER-END.
     EXIT.
* FIND OUT WHERE A STRING ENDS SO TRAILING BLANKS ARE NOT PRINTED
 FIND-END-OF-STRING SECTION.
     MOVE ZERO TO S2.
     PERFORM VARYING S1 FROM 1 BY 1 UNTIL S1 > 256
       IF STRING256 (S1:1) NOT = " " THEN
         MOVE S1 TO S2
       END-IF
     END-PERFORM.
```

```
 FIND-END-OF-STRING-END.
     EXIT.
     EJECT
* PROCESS SYSIN PARMS
 PARM-INPUT SECTION.
* SET ALL PARMS TO THEIR DEFAULT VALUES
     MOVE SPACE     TO WH3-QMGR
                       WH4-EVENT-QUEUE-NAME
                       W-USER-1
                       W-USER-2
                       W-USER-3
                       W-USER-4
                       W-USER-5.
     MOVE ON-VALUE  TO W-DETAILS-DEFINE
                       W-DETAILS-ALTER
                       W-DETAILS-DELETE
                       W-DETAILS-REFRESH.
     MOVE OFF-VALUE TO W-BROWSE
                       W-SKIP-USER
                       W-SKIP-OBJ.
     MOVE "ON"      TO WH5-DETAILS-DEF
                       WH5-DETAILS-ALT
                       WH5-DETAILS-DEL
                       WH5-DETAILS-REF.
     MOVE "GET"     TO WH7-BROWSE.
     OPEN INPUT SYSIN.
 PARM-INPUT-READ.
     READ SYSIN AT END GO TO PARM-INPUT-EOF.
     MOVE SPACE TO W-INPUT-KEYWORD W-INPUT-VALUE.
     UNSTRING SYSIN-DATA DELIMITED BY ALL SPACE INTO
         W-INPUT-KEYWORD
         W-INPUT-VALUE.
     EVALUATE W-INPUT-KEYWORD
         WHEN "QMGR"
            MOVE W-INPUT-VALUE TO WH3-QMGR
         WHEN "EVENTQUEUE"
            MOVE W-INPUT-VALUE TO WH4-EVENT-QUEUE-NAME
         WHEN "DETAILS-DEFINE"
            IF W-INPUT-VALUE = "OFF" THEN
               MOVE OFF-VALUE  TO W-DETAILS-DEFINE
               MOVE "OFF"      TO WH5-DETAILS-DEF
            END-IF
         WHEN "DETAILS-ALTER"
            IF W-INPUT-VALUE = "OFF" THEN
               MOVE OFF-VALUE  TO W-DETAILS-ALTER
               MOVE "OFF"      TO WH5-DETAILS-ALT
            END-IF
         WHEN "DETAILS-DELETE"
            IF W-INPUT-VALUE = "OFF" THEN
               MOVE OFF-VALUE  TO W-DETAILS-DELETE
```

```
               MOVE "OFF"      TO WH5-DETAILS-DEL
            END-IF
          WHEN "DETAILS-REFRESH"
            IF W-INPUT-VALUE = "OFF" THEN
               MOVE OFF-VALUE  TO W-DETAILS-REFRESH
               MOVE "OFF"      TO WH5-DETAILS-REF
            END-IF
          WHEN "BROWSE"
            IF W-INPUT-VALUE = "ON" THEN
               MOVE ON-VALUE   TO W-BROWSE
               MOVE "BROWSE"   TO WH7-BROWSE
            END-IF
          WHEN "SKIP-USER"
            UNSTRING SYSIN-DATA DELIMITED BY ALL SPACE INTO
                     W-INPUT-KEYWORD
                     W-USER-1 W-USER-2 W-USER-3 W-USER-4 W-USER-5
            MOVE     ON-VALUE      TO W-SKIP-USER
          WHEN "SKIP-OBJ"
            ADD 1 TO W-OBJSPTR
            IF W-OBJSPTR > W-OBJSMAX THEN
              MOVE     W-ERROR-12 TO W-SYSPRINT-DATA
              PERFORM  PRINT-LINE
              MOVE     W-OBJSMAX TO W-OBJSPTR
            ELSE
              UNSTRING SYSIN-DATA DELIMITED BY ALL SPACE INTO
                       W-INPUT-KEYWORD
                       W-OBJSKIPTYPE(W-OBJSPTR)
                       W-OBJSKIPNAME(W-OBJSPTR)
              MOVE     ON-VALUE      TO W-SKIP-OBJ
            END-IF
          WHEN OTHER
      * IGNORE LINES STARTING WITH "*"
            IF W-INPUT-KEYWORD (1:1) = "*" THEN
              NEXT SENTENCE
            ELSE
              MOVE    W-INPUT-KEYWORD TO WE1-KEYWORD
              MOVE    W-ERROR-1       TO W-SYSPRINT-DATA
              PERFORM PRINT-LINE
              MOVE    W-RC-WARNING    TO W-RETURN-CODE
            END-IF
       END-EVALUATE.
      * LOOP ON PARMLINE
       GO TO PARM-INPUT-READ.
      * END INPUT ON SYSIN
       PARM-INPUT-EOF.
       CLOSE SYSIN.
       IF WH3-QMGR = SPACE THEN
          MOVE    W-ERROR-2  TO W-SYSPRINT-DATA
          PERFORM PRINT-LINE
          MOVE    W-RC-ERROR TO W-RETURN-CODE
```

```
END-IF.
IF WH4-EVENT-QUEUE-NAME = SPACE THEN
   MOVE    W-ERROR-3  TO W-SYSPRINT-DATA
   PERFORM PRINT-LINE
   MOVE    W-RC-ERROR TO W-RETURN-CODE
END-IF.
IF W-RC-ERROR = W-RETURN-CODE THEN
   GO TO MAIN-END
END-IF.
```

Editor's note: this article will be concluded next month.

Stefan Raabe
Systems Programmer (Germany)

Exploiting WebSphere Business Integration Message Broker statistics for accounting purposes

This article describes an accounting function implementation that exploits WebSphere Business Integration Message Broker (hereafter called WebSphere BI MB) statistics traces. This accounting function is implemented as part of the product WebSphere Business Integration for Financial Networks Base (abbreviated to WebSphere BI for FN) for z/OS Release 2 and used with WebSphere BI for FN Extension for SWIFTNet (ESN).

INTRODUCTION

The purpose of the accounting function is to provide information about the usage of WebSphere BI for FN at customer installations. This information is needed for two reasons. Firstly, the product is priced based on the number and size of financial messages that are processed by the customer. WebSphere BI for FN is used by many IT service providers that process financial messages for internal and external customers. The service providers need the accounting information to charge their clients based on the number of

message processed by the client. This solution exploits the capabilities that are already available with the broker.

WebSphere BI for FN is an IBM product consisting of a Base and different extensions. The Base part provides functionality and services that can be used to more easily create and deliver products on top of WebSphere BI MB – for example message auditing and configuration services to dynamically influence the processing of message flows. The elements of the accounting function are added as part of WebSphere BI for FN Base for z/OS Release 2.

There are multiple product extensions that provide value-added functionality. For example, the extension for SWIFTNet provides access to the Secure IP Network, a private network operated by SWIFT. This extension supports different kinds of financial messages being sent and received and allows files to be transferred across the network.

The following sections describe the statistics capabilities of WebSphere BI MB and how this is exploited by WebSphere BI for FN.

MESSAGE BROKER STATISTICS OVERVIEW

With Version 5, WebSphere BI MB introduced functionality called accounting and statistics traces. The statistics and accounting function gathers dynamic information about the usage of message flows and processes details about the nodes that are involved in the processing of messages. Accounting and statistics information provides a customer with data about the usage of the broker that allows for chargeback for the invocation of message flows. Statistical data can also be used as the basis for monitoring and performance analysis tasks.

A message broker can provide different statistics:

- Message flow statistics.

  The statistics and accounting information for a message

flow includes information about the name of the flow, the name of the execution group a message flow is running in, the name of the broker, and information about the timeframe in which the data was gathered. In addition, it contains processing information, for example the number of messages processed in the measurement interval or the total, minimum, and maximum CPU time used to process the messages.

- Message flow thread statistics.

  Similar information as for a message flow is also collected for each thread that is processed for the message flow. For identification purposes, each thread is identified by a thread number.

- Node statistics.

  Node statistics contain information about all nodes within a message flow. For each node, information about its name, the number of input and output terminals and the node type is included. The node type is, for example, MQInputNode or ComputeNode. In addition, the statistics contain processing information, for example the number of times the node was invoked.

- Terminal statistics.

  For each terminal of a message processing node, the terminal statistics contain the name of the terminal, the information whether the terminal is an input or output terminal, and information about the number of invocations for this terminal.

The statistics and accounting information is gathered only if statistics and accounting is enabled. By default, the functionality is disabled. Collecting the statistics and accounting information is integrated in the broker. The broker must not be changed to get the information; also, a re-deploy of the message flows is not needed. A simple command can be used to turn the statistics on and off.

WebSphere BI MB provides two different types of statistics information – snapshot statistics and archive statistics. They can be used for different time periods. Each time period for each archiving type of statistics can be enabled and disabled separately and it is possible to enable both snapshot and archive statistics at the same time.

When snapshot statistics are enabled, the broker collects all information for a predefined short time interval and makes this information available. The time interval is roughly 20 seconds. It cannot be altered by a user.

Archive statistical data is designed more for accounting purposes and chargeback. This type of data is collected over a longer period. The default period is one hour, but it can be adjusted by a user in the range from 10 minutes to 10 days.

WebSphere BI MB can write the statistics and accounting information to different destinations. On all platforms the information can be written to the WebSphere BI MB user trace, and the broker can publish the information so that anybody interested in it can subscribe to the statistics and accounting information.

When writing the information to the WebSphere BI MB user trace, it must be handled like any other user trace information. It can be retrieved using the standard command **mqsireadlog** and formatted using the command **mqsiformatlog**. When using the user trace destination, tracing must not be active in parallel; if it is, the statistics and accounting information can be overwritten by the trace information. That's why this destination should mainly be used for short-time statistics information, eg when using the data for performance analysis. In this situation, usually a batch of messages is processed and afterwards the data is retrieved for analysis purposes. After that, the data is usually discarded.

The broker can also publish the statistics and accounting data. This is done using a predefined topic structure and the default subscription point. The topic structure looks like this:

```
$SYS/Broker/<broker name>/StatisticsAccounting/
                                        <type>/<EG name>/<Flow name>
```

The placeholders in this structure are substituted by the corresponding value for the broker name, the type of accounting and statistics information (which can be either SnapShot or Archive), the name of the execution group where the message flow is running, and the name of the message flow. Using this structure, an accounting application can subscribe to the information that it is interested in.

As can be seen from the topic name that starts with $SYS, statistics and accounting data publications are broker publications. The broker is always allowed to publish on these topics. If topic-based security for the broker domain is used, the user name that is running the application that requests the statistics and accounting information must be allowed to issue the subscription.

When publishing the information, the broker is issuing a publication with an RFH2 header and the accounting and statistics information in the message body. The message body is in XML format. The information is structured in a folder structure as shown in Figure 1. The details of each statistic are available as attributes to the appropriate element in the tree. Further information and details of the attributes can be found in the WebSphere BI MB documentation.

Publish and subscribe allows an application to get the accounting and statistics information for a whole WebSphere BI MB broker domain.

On z/OS the WebSphere BI MB broker offers a third destination for the statistics and accounting data. This destination is the Systems Management Facility (SMF). SMF is a standard z/OS high-performance service that can be used to store data, eg accounting and performance-related data. This facility offers all the necessary functionality to handle this kind of data, for example to switch to a different dataset if one gets full.

In order to get the information written to SMF, the SMF destination for the statistics and accounting information in the broker must be enabled in addition to being enabled in the

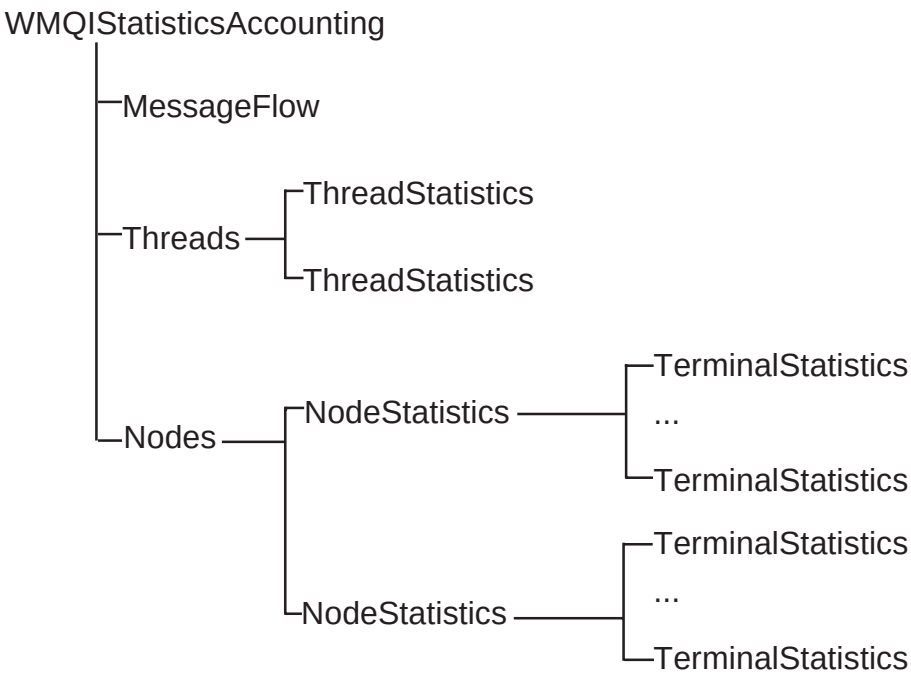

Figure 1: Statistics and accounting data XML structure

broker. The user ID of the message broker must be allowed to write the SMF data by permitting the user ID of the broker to the BPX.SMF facility class.

With WebSphere BI MB two additional commands are available to define and control the settings for statistics and accounting information. The command **mqsichangeflowstats** is used to define the setting. Using this command, the setting for one type of statistics and accounting information for one broker can be changed. The setting can be changed for a specific execution group or for all execution groups in the broker. Accordingly, the settings can address a specific message flow or all message flows in the specified execution group.

Furthermore, it can be specified what kinds of statistics are requested and to which destination the information should be written. The broker allows different destinations for snapshot statistics and for archive statistics.

If the statistics and accounting function is turned on, message flow statistics are always gathered. In addition, thread statistics and node statistics can be controlled separately. Terminal statistics can be included only if node statistics are also requested.

To display the current settings for statistics and accounting processing, the new command **mqsireportflowstats** is provided. The settings can be requested for one type of statistics and accounting information at a time. The requested information can be at the same granularity as with **mqsichangeflowstats**, either for a specific execution group or for all execution groups of the broker, and for a specific message flow or all message flows in the specified execution group.

Once enabled, the broker writes the statistics and accounting information to the specified destination. This is done every time the interval for the type of information expires. But there are also other events that can force statistics and accounting information to be written, for example, if message flows are re-deployed or if the broker is shutting down.

With CSD2 WebSphere BI MB introduced another capability to the statistics and accounting functionality. It allows the statistics information to be partitioned according to a customer-defined criterion. This criterion is the so-called Accounting Origin. To allow for partitioning of the statistics and accounting information, a message flow has to be modified to provide the Accounting Origin. This is done by providing a value in a field in the message environment, eg by using a compute node as shown below:

```
SET Environment.Broker.Accounting.Origin = 'MyDepartment';
```

The information in the environment field is recognized when

the flow returns to its input node. There is no pre-defined value that has to be entered into this field, but usually partitioning could be done according to the department or organizational unit the message belongs to.

ACCOUNTING WITH WEBSPHERE BI FOR FN

The accounting function of WebSphere BI for FN is based on the message broker statistics and consists of multiple parts, as shown in Figure 2.

The first part is in the message broker. WebSphere BI for FN contains many message flows. But only a few are relevant to accounting. The relevant message flows process the messages for any number of clients, represented as Organization Units (OU) in WebSphere BI for FN. To be able to segregate the

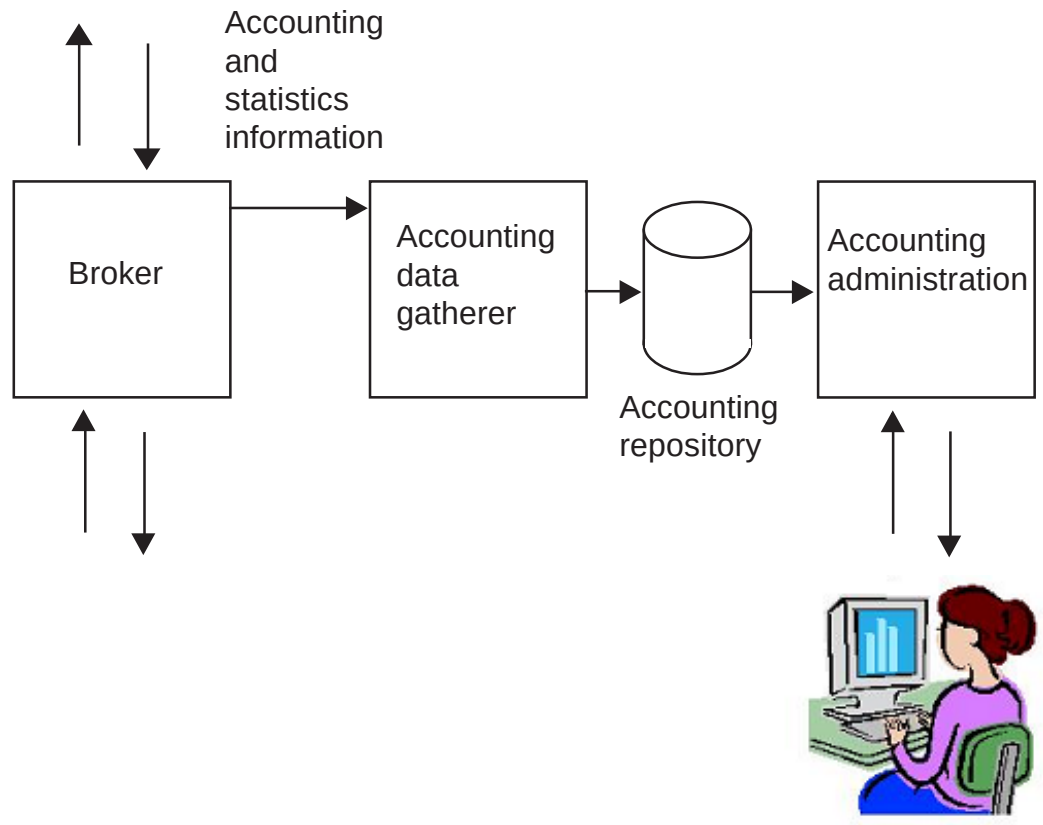

Figure 2: Accounting function overview

information for each OU, WebSphere BI for FN provides a node that can be inserted in those message flows that should be accounted. This node adds the information about the OU as one part of the message broker accounting origin into the environment of the message. Another part of the accounting information is derived from a property of the node. This property is called service type. The service type is introduced because, in most situations where the accounting function is used, there are at least two message flows that process the same kind of message. One message flow processes messages that should be sent via an external network, while another message flow processes messages that are received from the external network that is accessed through WebSphere BI for FN. Messages for both message flows should be accounted for in the same way. This can be achieved if both message flows use the same value for the service type property.

Once inserted, the broker collects the information, partitions it for each OU, and provides the statistics. As described above, the statistics have to be enabled for the relevant message flows. Enabling the flows is done as part of the deployment processing described by WebSphere BI for FN. For accounting purposes, WebSphere BI for FN requires archive statistics. This allows using snapshot statistics for other purposes, for example on-line monitoring.

The statistics information needs further processing. The information in SMF records and user trace are not immediately available. Therefore WebSphere BI for FN uses the destination XML. The message broker publishes the information as described above. A WebSphere BI for FN accounting data gatherer component subscribes itself to the corresponding topics. This component is implemented as a WebSphere BI MB message flow. This eases the processing of the information because parsing the published data structure is one of the basic functions of the message broker.

When processing in a message broker domain, there is one

such accounting data gatherer message flow on each broker, as shown in Figure 3.

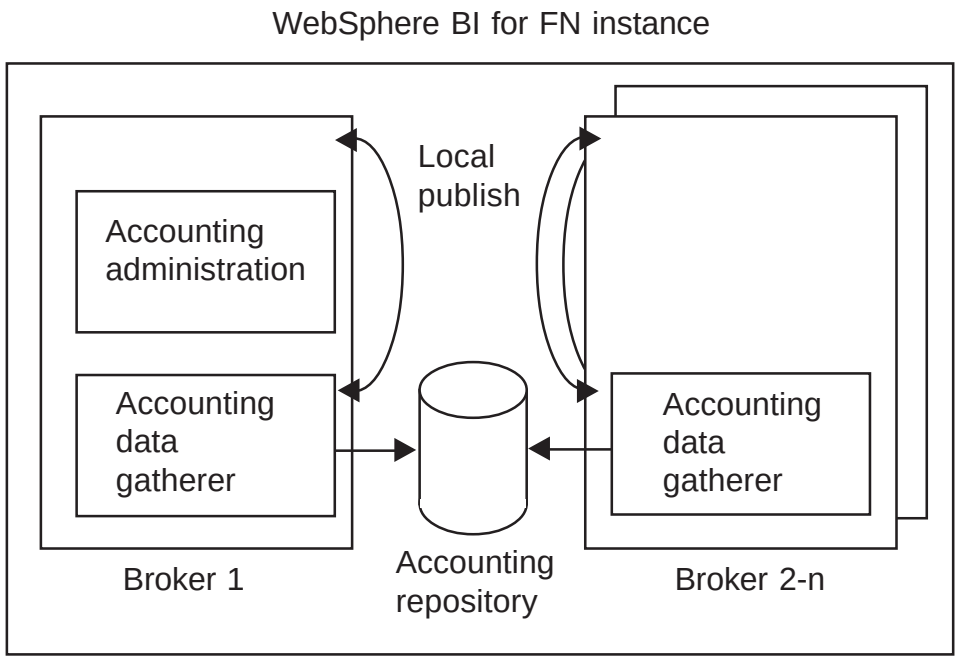

Figure 3: Accounting gatherer in a broker domain

In Figure 3, it can be seen that the accounting data gatherer registers only for local subscriptions. This ensures that the information is not processed multiple times, which would falsify the results. It also ensures that all messages are counted because there's no network between the publisher (the broker) and the subscriber (the accounting data gatherer) where the information could be delayed or lost.

The accounting data gatherer message flow extracts the information about the number of messages, the CPU costs for these messages, and the total size of all the messages. This information is contained in the message flow statistics – the minimum information provided when activating the statistics. This means the overhead for collecting the information can be minimized.

When extracted, the statistics information is stored in an

accounting repository. This repository is a database table in which the information for each message flow, OU, and service type is accumulated for each day. The reason for this granularity is that it provides flexibility when requesting information for different accounting periods, for example weekly, monthly, or quarterly.

Accessing the accounting repository using SQL is not very convenient for administrators. To ease administration of the accounting data, WebSphere BI for FN provides an accounting administration component. This consists of a message flow that performs all actions, and a command line program that accepts input from an administrator, formats this input into a WebSphere MQ message, receives the response from the message flow, and displays the results.

For administration purposes two actions are supported by the accounting administration message flow. These are a command to display accounting data and one command to delete old accounting data from the repository. Displaying accounting data can be done with different criteria to satisfy different usage scenarios. First of all, a query can be performed for all messages of a specific service type. This information is required to calculate the usage-based price of the product. The time span for which the information is requested is a parameter in the query. The time span to use depends on the usage conditions for the product. Having this flexibility allows the conditions for subsequent releases to be changed, or to have different conditions based on the different service types of the messages. The command returns the accounting information for all available service types. Optionally, the result can be restricted to a specific service type.

A service provider is more interested in the usage information for each OU. A parameter of the query command allows the information to be restricted to a specific OU. If the service provider needs the information for different OUs the administrator has to issue the query for each OU.

The result of the query command always contains the complete information collected for the time span. This is the accumulated

number of messages, the accumulated CPU time, and the accumulated size of all messages. Which criterion is used depends on the product extension or the contract between the service provider and their customers. Which schema is used is independent of the accounting functions. It's possible that the conditions for using the product and the way a service provider charges his clients are different. For example, ESN accounts financial messages by the number of messages and files by file size.

Accounting data is stored for every day and every service type. This structure means that the amount of data is growing each day. Therefore it is recommended that users delete old accounting data if it is no longer needed. When deleting old information from the accounting repository, data that is still needed for further accounting is not allowed to be removed. Therefore the command to delete the old accounting data has a parameter that specifies the date up to which the data is removed. How long the data should be retained may differ from customer to customer. Nevertheless, a minimum time period is required by the product itself. The command to remove old accounting data checks that no data from this time period is deleted.

SUMMARY

WebSphere BI for FN exploits WebSphere BI MB statistics information to collect information about the usage of its message flows in the broker. The information is maintained in an accounting repository. The information that can be retrieved from this repository is used for usage-based charging for the product. If the product is implemented by a service provider, the information can also be used for charging clients based on their usage of the product. Since the accounting functionality is delivered with the WebSphere BI for FN Base product, any customer or independent software vendor can use the functionality for his own message flows.

Michael Groetzner (michael_groetzner@de.ibm.com)
IBM (Germany)

mValent has announced Version 2.2.of Infrastructure Automation Suite, its configuration, change, and release management software for Web-based J2EE applications that use WebSphere or BEA WebLogic. The product helps users manage and automate the pre-production configuration process for Web-based applications as they pass from development to production.

mValent's software provides a configuration management database for the configuration files for the various applications involved, including dictionary entries with definitions, bounds, and templates, and a polling mechanism to periodically check current configuration file settings to determine whether changes have been made.

The new release adds 'enablers' that automate the management of disparate and interdependent configuration definitions.

For further information contact:
URL: www.mvalent.com/product/index.asp.

* * *

IBM has announced WebSphere Data Integration Suite, which is IBM's release of the integration platform it acquired from Ascential (who had code-named it Hawk). The software will include a new user interface, new metadata profiling, and enhanced linkage between Ascential's DataStage TX transactional and operational data-integration software and IBM's WBI Message Broker.

Future product integration plans include IBM providing a converged set of tools based on Eclipse for WBI and DataStage TX and a single repository architecture – including metadata discovery, exchange, and management – that will span existing IBM products.

For further information contact:
URL: www.ascential.com/products/eis.html

* * *

IBM and Cisco have announced they jointly plan to deliver speech-enabled self-service solutions that combine IBM's WebSphere Voice Server product and Cisco's Customer Voice Portal.

The solutions combines IBM's integration and application infrastructure software and speech technology with Cisco's Internet Protocol (IP) communications and focus on self-service speech applications, which together can enable easier deployment of customized speech applications that enhance the customer experience, the companies said.

By using IBM WebSphere Application Server middleware, the solutions would allow contact centres to use open standards, including Voice XML and J2EE.

For further information contact your local IBM representative.

* * *

Mainsoft has announced Visual MainWin for J2EE Developer Edition (Grasshopper), a freely available Visual Studio .NET plug-in, enabling Visual Studio developers to develop Web applications for Linux. Access to enterprise features such as multi-CPU capabilities and fully-featured J2EE servers, including WebSphere, JBoss, and WebLogic, are available in Visual MainWin for J2EE Enterprise Edition.

For further information contact:
URL: www.mainsoft.com/news/press_releases/2005_05_24_01.aspx.