Getting Started with Universal Command for SOA: MQ Connector 4.2.0

Objective

The objective of this document is to assist in the following activities regarding the Universal Command for SOA: MQ Connector:

- Installing Universal Command for SOA.
- Running Universal Command for SOA with an MQ Connector.

Installation Requirements

The following is required for running Universal Command for SOA with an MQ Connector:

- MQ Environment version 6 or later, with working queues.
- Stonebranch Solutions 3.2.0.0 or later; installed, licensed, and running.
- MQ Client jar files for native communication to MQ must be in the following path:

  /opt/universal/uac/container/webapps/axis2/WEB-INF/lib

  com.ibm.mq.commonservices.jar
  com.ibm.mq.jar
  com.ibm.mq.pcf.jar
  com.ibm.mq.headers.jar
  com.ib.mq.jmqi.jar
  connector.jar

Installation

These instructions describe installation of the Universal Command for SOA for AIX package.

Universal Command for SOA 4.2.0 is packaged as a compressed tar file.

The name of the Universal Command for SOA package file has the following format:

sb-soa-4.2.0.0-aix-5.2.tar.Z

(The name assumes product maintenance level 4.2.0.0 for Universal Command for SOA.)

The following steps describe the unpacking and installation of Universal Command for SOA 4.2.0:

1. Create a directory (or select an existing directory) in which to save the package file.
2. Save the package file into that directory.
3. Uncompress and extract the installation files in the current working directory. The command to extract the files is:

   zcat sb-soa-4.2.0.0-aix-5.2.tar.Z | tar xvf -

   If your operating system does not support the **zcat** command, use the following command:

   gunzip sb-soa-4.2.0.0-aix-5.2.tar.Z

   The output of the **gunzip** command provides the following **tar** file:

   tar –xvf sb-soa-4.2.0.0-aix-5.2.tar

4. After the extraction is complete, run the installation script, **upsinst**, which executes the **installp** command:

   ./upsinst

   An installation log is written to file **install.log** in the current directory. **upsinst** automatically restarts the Universal Broker daemon, **ubrokerd**, at the end of the install.

5. You can use the **uquery** command (cd to **/opt/universal/bin**) to validate that the 4.2.0 SOA for Universal Command is running:

   uquery -i localhost (or the name of your server)

   The output should have the following format:

```
Component ID.............: 1206554190
Component Name...........: uac (Server)
Component Description....: Universal Application Container Server
Component Version........: 4.2.0 Level 0 Development Build 284
Component Type...........: uac
Component Process ID.....: 27070
Component Start Time.....: 01:56:29 PM
Component Start Date.....: 03/26/2010
Component Command ID.....: uac
Component State..........: REGISTERED
Component MGR UID........:
Component MGR Work ID....:
Component MGR Host Name..:
Component MGR IP Address.:
Component MGR Port.......:
Component Comm State.....: STARTING
Component Comm State Time.: 01:56:29 PM
Component Comm State Date.: 03/26/2010
Component MGR Restartable.: NO
Component Comment........:
```

6. From the license file that was sent to you by Stonebranch, Inc., add the license information to the following file:

   /etc/universal/uacs.conf

7. Recycle **ubroker** using the following commands (cd to **/opt/universal/ubroker**)

   a. **./ubrokerd stop**

   b. **./ubrokerd start**

8. Again, you can use **uquery** (cd to **/opt/universal/bin**) command to validate that Universal Command for SOA 4.2.0 is running after updating the **uacs** configuration file.

   uquery -i localhost (or the name of your server)

MQ Environment Verification

Verify that you have a working MQ environment. You must define the following MQ values, as these are needed for the Universal Command for SOA jobs that you will submit: queuemanager, queuename, and channel.

You now can run jobs in MQ using the Universal Command for SOA: MQ Connector.

Running a Universal Command for SOA Job on z/OS Connecting to MQ Connector

1. Create the UCMD Manager JCL.

   This provides the UCMD Manager options, references to the MQ Connector options, and the payload.

   It has the following format:

```
//XXXXXXXX JOB CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
000002 //*
000003 //*************************************************************
000004 //*MQ queue test for Publish
000005 //*UCMD is the proc that calls UC Manager
000006 //*LOGON is the DD with userid and passwd (can use encrypted)
000007 //*SCR is the script that contains the MQConnector information
000008 //* to connect to an MQ Broker
000009 //*UNVIN provides the payload for the SCRIPT in SCR
000010 //*************************************************************
000011 //*
000012 //*       JCLLIB ORDER=LIB.V3207.UNV.UNVCONF
000013 //*
000014 //UCMD     EXEC UCMDPRC
000015 //LOGON    DD DISP=SHR,DSN=USER123.UAC.LOGON(USER)
000016 //SCR      DD DISP=SHR,DSN=USER123.UAC.SCR(MQPUB)
000017 //UNVIN    DD DISP=SHR,DSN=USER123.UAC.PYL(MQPYL)
000018 //UNVOUT   DD SYSOUT=*
000019 //UNVERR   DD SYSOUT=*
000020 //SYSIN    DD  *
000021 -s scr
000022 -script_type SERVICE
000023 -i ucaserver -f logon
```

2. Create the MQ Connector Command Options Data Set Member.

   This member contains the command options for the MQ Connector that specifies the required information to submit a job to the MQ environment.

   It is referenced with the **SCR** ddname and has the following format:

   ```
   -protocol mq
   -mep Publish
   -mqhost MQHOST
   -mqqueuemanagername MyQueueManager
   -mqqueuename UpsQaQueue
   -mqchannel UpsQaChannel
   -timeoutsec 120
   ```

3. Create the Payload Data Set Member.

   This member contains the MQ message and is read in via STDIN.

   Note: The **LRECL** length depends on the job it describes. Verify that your data set member record length can accommodate the maximum line length of your message.

   Example:

   ```
   000001 <?xml version="1.0" encoding="UTF-8"?>
   000002 <Message>Hello...this is a payload in a MQ message.</Message>
   ```

Running a Universal Command for SOA Job on UNIX Connecting to MQ Connector

1. Create the UCMD script file (**Mqopt**) to contain the option parameters.

   Mqopt contains:

   ```
   -protocol mq
   -mep Publish
   -mqhost MQHOST
   -mqqueuemanagername MyQueueManager
   -mqqueuename UpsQaQueue
   -mqchannel UpsQaChannel
   -timeoutsec 120
   ```

2. **MQPayload.xml**

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <Message>Hello...this is a payload in a MQ message.</Message>
   ```

3. From a command prompt, execute the following command to send a message to an MQ Queue:

   ```
   ucmd -script  Mqopt -script_type SERVICE -i ucaserver -u user
   -w user < MQPayload.xml
   ```

   You can also execute the command using the Universal Command options for STDIN (**-I** for input and **–F** for file):

   ```
   ucmd -script  Mqopt -script_type SERVICE -i ucaserver -u user
   -w user -I -F MQPayload.xml
   ```

4. Example output:

   ```
   MQ message published successfully on destination UpsQaQueue.
   ```