Volantis Mobility Server

Multi-Channel Server™ ver. 5.1

Installation Guide

Volantis Mobility Server

Multi-Channel Server™

Version 5.1

Installation Guide

Multi-Channel Server: Installation Guide

Version 5.1
Published date 2009-01-29

Included software

Apache Software Foundation
Free Software Foundation
Jaxen Project
Sun Microsystems Inc

Table of contents

Installing MCS

Installing and configuring Multi-Channel Server (MCS) involves a number of steps, outlined here, and detailed in related topics.

Media Access Proxy and Message Preparation Server are optional installations. MAP is required if you wish to transcode media to match device characteristics. MPS is required if you wish to use push messaging with MCS.

Volantis Mobility Server includes the *Getting Started* guide, which contains instructions for compiling the open source code, and quickly installing a working development system with Eclipse and Tomcat. *Getting Started* is available in both XHTML and PDF formats on the community documentation site.

Note: If you have an existing MCS repository, you should ensure that it is fully backed up prior to any maintenance work.

Required software

MCS operates as an application within a web application server. The web application server you choose must be installed and working successfully before MCS can be used.

At design time, you define policies for devices, layouts, components and themes in an XML repository. The policy editors operate within the Eclipse Framework as a set of integrated plugins. The Eclipse Framework needs to be downloaded and installed before the policy editors can be used.

At runtime, the XML policy repository may also be deployed as a set of tables within a SQL database. If you wish to use a database as a repository you should have the client installed and working before you install MCS.

Running the installer

The installation wizard steps you through a series of pages containing settings that correspond to the entries in the `mcs-config.xml` file.

Setting up your runtime repository

The MCS installation will create a `devices.mdpr` file that contains the MCS device repository information. You can access the repository with the Device Repository editor. Additionally, individual XML files are created as policies, and are constructed and modified in a default project.

If the policy repository needs to be deployed within a SQL database at runtime; appropriate users and tables will need to be created in the database management system, and any required XML policies will need to be imported into the database.

Completing the setup

The MCS configuration file is used to control the operation of MCS. However, the initial settings created by the installation process will normally be adequate.

This section deals with any other tasks required.

Installing MAP

MCS relies on the Media Access Proxy (MAP) which also incorporates the ICS servlet for URL and STI transcoding services. MAP has a separate installer. When you have completed your MCS installation, you should install and configure MAP and ICS.

Verifying the MCS installation

Finally, you will want to be reassured that everything works. A sample 'welcome' JSP page is available to test your MCS installation.

Note: Because file locations are set during installation, the configuration examples shown in related topics use placeholders for parts of the directory structure. You should substitute `[path]` with the path preceding the location described, and `[context_root]` with the application root context.

Required software

MCS requires a repository which stores MCS policies for device, digital asset and stylistic information. This is implemented at design time as a set of XML files. If you wish to implement a database repository for runtime deployment, you must have a JDBC or relational database client installed, configured and working correctly before you can use the MCS software.

A web application server needs to be installed and working before you can view any MCS generated pages. You may generate the pages without the web application server being installed but you will not be able to view them. You can also use the policy editors, to view and update the repository, without a web application server being installed.

If you want to use MCS with a web application server, the web application server should be installed and configured before you install the MCS software. This will allow the MCS configuration file to be updated correctly by the installer, avoiding the need to set the configuration for the web application server manually.

Note: A full list of supported platforms for development and runtime environments is contained in *Supported platforms* in *MCS Release notes* (PDF).

Eclipse framework

The MCS policy editors operate within the open-source Eclipse Framework.

Download and install a supported version of Eclipse, following the instructions from *Eclipse downloads*. Ensure that you install the Release Build SDK for your chosen platform.

A set of plugins for the MCS policy editors are created during installation. These will need to be copied to the appropriate Eclipse plugins directory. See *Completing the installation* for details.

Database server

The database must be available to any computer on which the MCS server runtime is installed. You can install a database on the same computer as the MCS components, or on a different computer.

Your database server should be installed with the appropriate services started. Refer to the appropriate database documentation for all products.

Web application server

A wide variety of web application servers can be used with MCS. *Setting up your web server* contains several examples.

Environment variables, as described in the installation guide for each web application server, will need to be set before installation.

If you have the JDK or JRE already installed you can test your web application server using the test instructions supplied with the server.

Refer to the appropriate application server documentation for all products.

Running the installer

The installation wizard steps you through a series of pages containing settings that correspond to the entries in the `mcs-config.xml` file. You can edit this file at any time after installation if you need to change the settings.

Most of the installer values have defaults. When you have completed a section, click **Next** to move to the next page, or click **Previous** to review or modify any values.

When installation is complete you can choose to save an automated installation file at a convenient location. Then you can repeat the installation on another machine with the path to the configuration file as a parameter.

1. Start the installer with the command
   `java -jar mcs_installer-5.1.jar`
2. Accept the license agreement
3. Accept the default feature selection or clear the check boxes and select individual components for installation
4. Choose a JDBC database vendor or specify an XML repository. If you choose 'XML' in the dropdown list you will skip to step 6.
5. Enter the database connection details. If you are migrating a database repository you should create a new user. Remember to include the default *Project* name to be used in the database.
6. Choose a web application server and set its configuration details
7. Accept the default Media Access Proxy (MAP) *URL* or enter an alternative

   Note: MAP must be installed separately. See *Installing MAP* for details.
8. Click **Browse** to choose the install location or accept the default
9. The installer displays a summary of some installation choices. Use the **Previous** button to go back and change any options. When you are satisfied with all your choices, click **Next** to start the installation.
10. Optionally save your installation settings by clicking **Generate an automatic installation script**, and naming the file. To repeat the installation run the following command.
    `java -jar mcs_installer-5.1.jar [configuration_file]`
11. Click **Done** to complete the installation

Setting up the MCS runtime

At design time MCS users develop policies in an XML repository. These XML policies can be used for runtime deployment, or they can be implemented within a set of tables in a relational database.

If you selected the XML repository type in the MCS installation wizard, MCS will be automatically configured to use the XML repository.

If this is a new installation and you are creating a new repository, you will need to create the tables holding the repository information under a specific user. This involves:

- Setting up appropriate JDBC driver jar files for database access
- Creating a new database user and tables

Note: Throughout this section, you should also refer to relevant documentation for your database.

Setting up JDBC driver jar files

If you are using a database to hold your repository, you will need to make sure that MCS can access any JDBC driver jar files correctly. You will need a `CLASSPATH` environment variable entry for your database client, and copy the corresponding jar files to the `MCS/[context_root]/WEB-INF/lib` directory. Check the database documentation for details.

For example, if you are accessing Oracle from a Windows machine, modify or setup a system environment variable called `CLASSPATH` to reflect the location of the corresponding jar file such as `C:\jdbcdrivers\classes1x.jar`. From a Linux machine , this might be `/home/jdbcdrivers/classes1x.jar`.

Note: The x in the filename will be another numerical digit. Refer to the valid filename in the `%ORACLE_HOME/jdbc/lib` directory.

You will also need to copy any jar files to the `MCS/webapps/[context_root]/WEB-INF/lib` directory. This also makes sure they are included correctly for the setup of your application server.

Note: If you are using Weblogic 7 and an Oracle database, there may be a `classes1x.zip` file under the `weblogic/common/lib` or `weblogic/server/lib` directories. If so, then move this file elsewhere so that the correct oracle JDBC driver file is picked up at runtime from the `MCS/webapps/[context_root]/WEB-INF/lib` directory which is deployed as a WebLogic application.

Creating a new database user

You will need to create a user with administrator privileges that matches the user and password you specified during installation. Both have default installation values of "mariner". These user and password values are also automatically set in the `mcs-config.xml` file.

Refer to the appropriate documentation for your database and your database administrator to create an appropriate new user.

Note: If you create the tables under a different user to the name you specified during installation then you will need to change entries in the `mcs-config.xml` file that refer to the database user name and password (under your application server's `MCS/webapps /[context_root]/WEB-INF` directory).

Creating new database tables

Specific example scripts to create or drop the repository tables are provided for Microsoft SQLServer along with generic SQL92 versions for use by Oracle and other databases.

Connect to the database as the user you have just created and run the appropriate version of one of following SQL scripts:

- `create_vm_tables.sql` - sets up tables for use by a single database user
- For Oracle and other SQL92 databases, use the scripts under your installation in `MCS/repository/jdbc-repository/sql92`

  Note: The directory `MCS/repository/jdbc-repository/oracle` contains an Oracle script for creating new user, and should not be used.
- For Microsoft SQLServer, use the scripts under your installation in `MCS/repository/jdbc-repository/mssql`

Importing XML policies into the database tables

You create the contents for the database tables by importing the XML policy files with the **mcsImport** command. These policies could be from a migrated repository, or created with the policy editors that need to be deployed for database access at runtime. See *Importing and exporting repositories* for details of the command parameters.

If you wish to import the policies and device repository to test the installation using the `welcome.jsp` page, you can access the associated XML policy files created during the installation as follows.

The policy files are under the `/MCS/repository/xml-repository` directory. The `devices.mdpr` file is under the `/MCS/repository/device-repository` directory. Refer to the *MCS Administration Guide* for command line parameters.

Note: This stage is not needed if you chose an XML runtime repository during installation, as the MCS configuration file will be set up to automatically access the policies and devices file under the `xml-repository` and `device-repository` directories.

Setting up your web server

After you have installed MCS, you need to set up the MCS server, essentially the contents of the `MCS/webapps/[context_root]` directory, as a web application for your application server. This directory also includes an example `welcome.jsp` application for testing and verification purposes.

The web application servers which are compatible with MCS are listed in *Supported platforms* in *MCS Release notes* (PDF). The related topics describe examples for specific web application servers.

Note: You should also refer to the documentation for your application server.

Installing on Apache Tomcat

You need to create a directory that the machine running Tomcat can access, to contain the MCS installation `MCS/webapps/[context_root]` directory. It can be local to the server machine or accessed via a drive mapping. For example it might be a directory under the `webapps` directory of the Tomcat installation.

1. Create a directory named `[context_root]` under the Tomcat `webapps` directory
2. Copy the entire contents of the `MCS/webapps/[context_root]` directory to the Tomcat webapps/[context_root] directory. For example, **$CATALINA_HOME/webapps/[context_root]**

Note: If you are using Tomcat on a non-windows machine, copy the `xercesImpl-2.6.2.jar` and `xmlParserAPIs-2.6.2.jar` files from `MCS/lib` directory to the `$CATALINA_HOME/common/lib` directory. Make sure that the generic `xerces.jar` file is not in the `$CATALINA_HOME/common/lib` directory and is copied elsewhere. This is to ensure that the correct version of Xerces parser is used.

Installing on IBM WebSphere

It is assumed that WebSphere is already installed, for example at `/opt/WebSphere/AppServer`.

You need to create a `.war` file to contain the contents of the `[path]/MCS/webapps/[context_root]` directory, and install a new WebSphere application that references the `.war` file.

1. In the directory above `[path]/MCS/webapps/[context_root]`, create a `.war` file with the command **jar -cvf ../[filename].war**.

   Note: The space and the full stop after the filename at the end of the jar command will ensure that you capture the contents of the current directory.
2. Log in to the WebSphere Administration Console at `http://localhost:9090/admin`
3. Expand the *Applications* option
4. Choose the *Install Applications* option
5. Select local path and click **Browse** to locate the `.war` file you created
6. Set the context root to `[install` directory and click **Next**.
7. Accept the default settings of 'do not override existing bindings' and 'default_host'. Click **Next**.
8. Accept the default settings of 'distribute applications' and 'create Mbeans for resources'. Your `.war` file now has '_war' at the end instead of a '.war' extension. Click **Next**
9. Choose the *MCS* web module option and click **Next**
10. Choose the *MCS* web module again. Click **Next** to display the installation summary.
11. Click **Finish** if you are satisfied with the entries. When the installation has successfully completed you will need to save your Master configuration changes.
12. Choose the *Enterprise Applications* option, click on the `.war` file you just installed and choose the Configuration tab
13. Change ClassLoader Mode to PARENT_LAST
14. Change WAR ClassLoader Policy to Application
15. Apply your changes, save the configuration and restart the server

Installing on BEA WebLogic

It is assumed that WebLogic is already installed, for example at `/opt/bea/weblogic`.

You need to copy the contents of the `[path]/MCS/webapps/[context_root]` directory to the `weblogic/server/bin` directory and configure a new WebLogic web application that references this directory.

> **Note**: WebLogic allows you to deploy `.ear`, `.jar` and `.war` files, but only exploded directory deployment is detailed here, as it is useful for a development environment. Refer to the WebLogic documentation for details of `.ear` and `.war` deployment.

1. Create a web application directory, for example `/opt/bea/weblogic/server/bin/[context_root]`
2. Copy all the files and directories from your MCS installation `webapps/[context_root]` to the new directory
3. Start the WebLogic server, for example **/opt/bea/weblogic/server/bin/startWLS.sh**
4. Log in to the Web Administration Console `http://hostName:7001/console`
5. Click the *Web Applications* option and choose **Configure a new Web Application**
6. Scroll to Step 2 and choose the *select* option next to the web application you have just created
7. In step 3, select a server from the Available Servers list and move it to the Target Servers list
8. In step 4, enter a name for the application or use the default
9. In step 5, choose **Configure and Deploy**

You will be presented with the Deploy tab with an indication of the deployment status , wait for the deployed status to indicate True rather than False

You can now verify your installation.

> **Tip**: Starting up under BEA WebLogic causes a `ClassCastException` trying to initialize the SAX parser. This is a known WebLogic issue, which can be fixed by upgrading to the latest patch version.

Creating a connection pool and datasource

The default during installation is to use a connection pool within MCS and the entries in the `mcs-config.xml` file are set up to enable this. This example shows how to set up a connection pool in WebLogic using the console.

> **Note**: If you choose to set up database connection pooling, you will need to remove or comment out the default connection pool element under the jdbc-repository section in the `mcs-config.xml` file. See the *Local repository configuration* for more details.

1. Open the Web Administration Console, for example `http://myHost:port/console/`
2. Click the *Connection pools* option under the JDBC section
3. Click *Configure a new connection pool* and enter appropriate details, for example for an Oracle database thin driver:

   Name - sample
   URL - jdbc:oracle:thin:@MyHost:1521:MySource
   Driver Classname - oracle.jdbc.driver.OracleDriver
   Properties - user=myuserid password=password dll=ocijdbc8 protocol=thin
   ACLName - if you have one, see the Security section of Web Administration Console.
   Click **Create**

4. Go to the Connections tab and configure the JDBC connections as required
5. Go to the Targets tab. Select the server you created that you wish to use with the connection pool, move it to the chosen list, and click **Apply**.
6. Go back to the WebLogic Console home
7. Click the *DataSources* option under the JDBC section
8. Click *configure a new JDBC data source*, and enter appropriate details; for example for an Oracle database

   Name - mydatasource
   JNDI Name – mydatasourceDS
   PoolName - sample
   Click **Create**

9. Click the Targets tab. Select the server that you wish to use with the datasource, move it to the chosen list and click **Apply**.

> **Note**: After creating a connection pool and a datasource you need to restart the server to reflect the connections.

Installing on JBoss

You need to create a .war file to contain the contents of the [path]/MCS/webapps/[context_root] directory, and copy this file to $JBoss/server/default/deploy.

1. Create a .war file in the [path]/MCS/webapps/ directory
   ```
   jar -cvf [context_root].war *
   ```
2. Copy this newly created file to the $JBoss/server/default/deploy directory
3. Start the server

Make sure that your jboss-service.xml file contains the following settings. That is to ensure the environment is compatible with the Java Servlet specification.

```
<attribute name="Java2ClassLoadingCompliance">false</attribute>
<attribute name="UseJBossWebLoader">false</attribute>
```

Completing the installation

Configuration files

The `mcs-config.xml` configuration file defines system parameters for MCS. The file is found in the `[path]/MCS/webapps/[context_root]/WEB-INF/` directory. The installation process will have set the basic parameters and you should have no need to change these until you have become more experienced in the use of MCS. A description of `mcs-config.xml` and how to use it can be found in *Configuring MCS* in the Administering MCS section of Help.

Support JARs

Support JARs are shipped with MCS. These should only be used when instructed by as they are for debugging support purposes. If used they will output debug logging information that will have a detrimental effect on the performance of MCS.

Setting up the XML Parser

MCS uses the Xerces parser by default. If you wish to use this XML parser you need do nothing further. If you decide to use any other XML parser you must use the system property org.xml.sax.driver to specify the location of the SAXParserFactory class.

Setting up the policy editors

After you have installed MCS, you need to set up the plugins for the policy editors in the Eclipse framework.

You should copy the contents `/MCS/eclipse/plugins` directory, including any sub-directories, to the plugins directory location, for example `/opt/eclipse/plugins/`. You should also copy the contents of `/MCS/eclipse/features/` to the features location, for example `/opt/eclipse/features/`.

When you launch Eclipse, these will be available as a set of wizards and editors for creation and manipulation of MCS objects.

Setting up the Media Access Proxy (MAP)

If you installed MAP, you can learn about its features and configuration in *Media Access Proxy* in the MAP section of Eclipse Help.

Verifying the installation

After installing the MCS application, setting up your application server and database if required, and completing any post-installation tasks you can now verify that your installation has been successful.

Displaying the welcome page

You can verify that your installation is working by displaying one of two welcome pages created during the web application server setup, `welcome.xdime` and `welcome.jsp`. The files are in the `/MCS/webapps/mcs/welcome` directory on your application server.

> **Note**: If you chose to use a database for runtime deployment, the appropriate policies need to exist within the database tables.

Start the web application server and MCS and load a welcome page into a Web browser. If a page appears your installation is working.

For example for a Tomcat installation, from a command prompt, navigate to the bin directory and issue the **startup** command. The server will report its usual startup messages. This will also mean that the MCS server is effectively started and available.

Start a web browser and enter a URL that references the web application instance, the directory and the welcome page. For example, if you are running the browser on the same machine as the server, you might use `http://localhost:8080/[context_root]/welcome/welcome.xdime`

Accessing the policy editors

You need to start Eclipse to access the policy editors.

You can then choose one of the MCS wizards for creating policies by choosing **File** | **New** | **Other** | **MCS**. See *Choosing a Wizard* for further information.

If you wish to review the policies used by the welcome pages you can import them into an MCS project in Eclipse.

To create a project see *Creating a project* in MCS Help.

To import the welcome policies into the project:

1. Expand the project and open the empty folder `mcs-policies`
2. Right-click and choose **Import** and then **File System** and **Next**
3. Browse to pick up the policy definitions in the XML repository, for example `/MCS/repository/xml-repository`
4. Click **Select all** and choose the *Create selected folders only* option
5. Click **Finish** to import the files

If you now expand the mcs-policies directory you will now see the imported policies for both the tutorial and welcome applications.

> **Note**: You can associate projects with the supplied device repository file `devices.mdpr` which is located under the `/MCS/repository/device-repository` directory.

If you have problems

As MCS executes, by default it writes log and status information to a file named `.log`. The file will be written to the log directory specified in the `web.xml` file.

References

Apache Software Foundation

    http://apache.org

Free Software Foundation

    http://www.fsf.org

Jaxen Project

    http://jaxen.org/

Sun Microsystems Inc

    http://www.sun.com

Eclipse downloads

    http://www.eclipse.org/downloads/