# TIBCO® MDM

# Installation and Configuration Guide

*Software Release 9.2*
*September 2019*

## Important Information

This software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. Please see the readme.txt file for the availability of this software version on a specific operating system platform.

THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "READ ME" FILES.

This and other products of TIBCO Software Inc. may be covered by registered patents. Please refer to TIBCO's Virtual Patent Marking document (https://www.tibco.com/patents) for details.

# Contents

# Figures

# TIBCO Documentation and Support Services

### How to Access TIBCO Documentation

Documentation for TIBCO products is available on the TIBCO Product Documentation website, mainly in HTML and PDF formats.

The TIBCO Product Documentation website is updated frequently and is more current than any other documentation included with the product. To access the latest documentation, visit https://docs.tibco.com.

### Product-Specific Documentation

Documentation for TIBCO MDM is available on the TIBCO MDM Product Documentation page.

Documentation for TIBCO products is not bundled with the software. Instead, it is available on the TIBCO Documentation site. To directly access the documentation for this product, double-click the following file: `TIBCO_HOME/release_notes/TIB_mdm_versionnumber_docinfo.html`

where *TIBCO_HOME* is the top-level directory in which TIBCO products are installed. On Windows, the default *TIBCO_HOME* is `C:\tibco`. On UNIX systems, the default *TIBCO_HOME* is `/opt/tibco`.

The following documents for this product can be found on the TIBCO Documentation site:

- *TIBCO MDM Release Notes*
- *TIBCO MDM Installation and Configuration Guide*
- *TIBCO MDM User's Guide*
- *TIBCO MDM System Administration*
- *TIBCO MDM Customization Guide*
- *TIBCO MDM Workflow Reference*
- *TIBCO MDM Web Services Guide*
- *TIBCO MDM API Reference*
- *TIBCO MDM Best Practices Guide*
- *TIBCO MDM Performance Tuning Guide*
- *TIBCO MDM Troubleshooting Guide*
- *TIBCO MDM Security Guidelines*

### How to Contact TIBCO Support

You can contact TIBCO Support in the following ways:

- For an overview of TIBCO Support, visit http://www.tibco.com/services/support.
- For accessing the Support Knowledge Base and getting personalized content about products you are interested in, visit the TIBCO Support portal at https://support.tibco.com.
- For creating a Support case, you must have a valid maintenance or support contract with TIBCO. You also need a user name and password to log in to https://support.tibco.com. If you do not have a user name, you can request one by clicking Register on the website.

### How to Join TIBCO Community

TIBCO Community is the official channel for TIBCO customers, partners, and employee subject matter experts to share and access their collective experience. TIBCO Community offers access to Q&A forums, product wikis, and best practices. It also offers access to extensions, adapters, solution accelerators, and tools that extend and enable customers to gain full value from TIBCO products. In addition, users can submit and vote on feature requests from within the TIBCO Ideas Portal. For a free registration, go to https://community.tibco.com.

# Installation Overview

Download TIBCO MDM from https://edelivery.tibco.com. To login, you need user name and password. If you have not received a user name and password, contact TIBCO Technical Support. After you download TIBCO MDM, install it using the installer provided.

### Default Installation Directory

- **Microsoft Windows**
  The default installation location is `$TIBCO_HOME` where all TIBCO products are installed. Typically, `$TIBCO_HOME` is at `c:\tibco`.
- **UNIX**
  The default installation directory depends on who performs the installation:
  - For root users, the default installation directory is /opt/tibco.
  - For non-root users, the default installation directory is /*myhome*/tibco, where *myhome* is the home directory of the user.

### Installer Disk Space Requirements in Temporary Area

- **Microsoft Windows Platforms**
  The entire package is extracted into a temp folder (minimum requirements 40 GB and 4 MB RAM), typically SystemDrive:\Temp or SystemDrive:\Documents and Settings*user_name*\Local Settings \Temp.
- **UNIX Platforms**
  The installer launcher first extracts a Java Virtual Machine (JVM) in a temporary directory (minimum requirements 40 GB and 4 MB RAM) and uses this JVM to launch itself. The size of the extracted JVM differs from platform to platform. You can select the temporary area using the following option when starting the installer:
  *install_package_name*.sh -is:tempdir /*temp_area*

## Prerequisites for Installation

Before you start the *TIBCO MDM* installation, ensure that your system meets all of the requirements.

*Prerequisites*

| Software Component | Description |
|---|---|
| **JDK** | TIBCO MDM is not certified with Open JDK. For the WildFly application server, TIBCO MDM supports Java 11 as a runtime environment. Starting from the 9.2.0 release, Java is bundled with the installer. You do not need to download Java from the Oracle download site. If any updates to Java 11, TIBCO MDM continues to ship through the hotfix installers.<br>For the WebLogic and WebSphere application servers, TIBCO MDM supports Java 8. To install JDK, download the executable file from the Oracle web site, run it and follow the instruction screens. For open Java, if you come across any TIBCO MDM problems that require support, download and point to the Oracle release (*JAVA_HOME*) to verify that the issue is reproducible before contacting TIBCO support.<br>On Windows, JDK is installed in the *install_directory*\Program Files\Java\ directory. The *install_directory* is the directory where Windows is installed. Though a typical JDK installation sets the *JAVA_HOME* environment, ensure that the variable has been set correctly.<br>Consult the readme shipped with your installation of TIBCO MDM for the most up-to-date software requirements. |
| **JMS Server** | The JMS Server must be installed and running with the required queues and topics created |
| **Application Server - For Typical install only** | The Application Server must be installed and running with the correct service packs applied.<br>For IBM WebSphere, make sure that JDK patch level matches the application server fix pack level. |
| **Database - For Typical install only** | The Database server must be ready with either Oracle, PostgreSQL, or SQL Server installed and must have a user account with full privileges for the database. It is also recommended that a second user be created, but with restricted privileges. |
| **Client - For Typical install only** | The client for the database must be installed on the TIBCO MDM system machine and must have access to Java JDBC connectors. The SQL Server client is required for creating new seed data. However, we do not need client for PostgreSQL database.<br>Oracle Client Software should be Developer Edition or Enterprise Edition and must be on the computer hosting the application server. TIBCO MDM uses the sqlldr utility shipped with these Oracle Client Software editions. |
| **Web Server - For Typical install only** | Web server is optional and is needed only if you do not plan to use direct URL access to application server. if Web server is going to be used, install it first. |

| Software Component | Description |
|---|---|
| **Cache Server - For Typical install only** | Cache server is optional and is needed only if a centralized cache sever is proposed. Make sure cache server is installed (but not running).<br>**Note**: For information on the Cache server, refer, Configuration of TIBCO MDM with TIBCO ActiveSpaces. |

## Additional Software Components

TIBCO MDM requires additional software components as listed in the following table. The requirements of components depend on your installation choices and supported platforms. For a complete list of versions and platforms supported, refer to the *Readme.txt* file.

*Required Components*

| Component to Install | Supported options | For more information, refer: |
|---|---|---|
| JDK | • For JBoss WildFly, WebLogic, and WebSphere Application Servers, use `1.8` version | • Installation Overview |
| **Database**<br>Install and configure a database. | • Oracle<br>• Microsoft SQL Server<br>• PostgreSQL | • Configuration of Oracle Database<br>• Configuration of SQL Server Database<br>• Simple Installation with PostgreSQL |
| Database Client | • Oracle<br>• Microsoft SQL Server | |
| **JMS Server**<br>Configure a JMS Server | • TIBCO Enterprise Messaging Service | • Creating TIBCO EMS Queues |
| **Application Server**<br>Configure a supported Application Server. | • Websphere with or without Websphere ND<br>• Weblogic<br>• JBoss WildFly | • Configuring TIBCO MDM for WebSphere Application Server<br>• Configuring TIBCO MDM for WebLogic Application Server<br>• Configuring TIBCO MDM for JBoss WildFly Application Server |

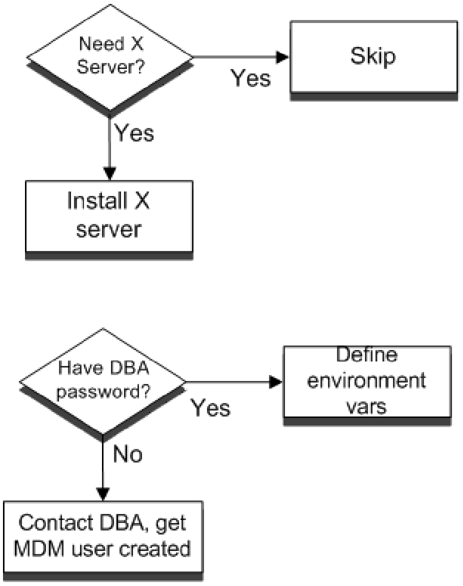
*Optional Components*

| | Component to Install | Supported options | For more information, refer: |
|---|---|---|---|
| 1. | **AS2 Gateway** Configure AS2 Gateway for secure communication with other systems. (required only for GDSN mode or external communication). | - Any AS2 Server<br>(such as TIBCO BusinessConnect) | Configuration of TIBCO BusinessConnect and TIBCO BusinessWorks<br>Details for configuring TIBCO BusinessConnect. Refer to appropriate documentation for other gateways. |
| 2. | **Cache Server** Configure a cache Server. | - TIBCO ActiveSpaces®<br>- Apache Ignite | - Configuring TIBCO ActiveSpaces<br>- Configuration Properties of Apache Ignite |
| 3. | **Web Server** Configure a supported Web Server to connect to the application server. | - IBM HTTP<br>- Apache Server<br>- Microsoft IIS | Configuration of Web Servers |
| 4. | **X Server** Configure to upload images for any records maintained using TIBCO MDM. | RealVNC | Installing X Server |

# Installing X Server

X Server should be running, if you plan to upload images for any records maintained using TIBCO MDM. TIBCO MDM uses the X server's rendering buffer to resize an image when it is uploaded. The *`DISPLAY`* environment variable is used in conjunction with the X server.

*X Server*



One popular X server is RealVNC (http://www.realvnc.com/). Some UNIX distributions come with a bundled X server or with the VNC X server pre-installed.

For example, to start VNC on a machine running Linux,

### Procedure

1. Run the following VNC server command: **`[vsadmin@hqstage01 vsadmin]$ vncserver`**
   A message is displayed to create a password, which is required to access your desktop.
2. Type the password.
   The following message is displayed: `New 'X' desktop is hqstage01.tibco.com:1`
3. Set the *`DISPLAY`* environment variable as follows: export `DISPLAY=hqstage01.tibco.com:1.0`

   If image upload does not work even though the VNC Server is running, add the following property in the generic JVM arguments:
   `-Djava.awt.headless=true.`

   - Default startup script is available in: `/home/vsadmin/.vnc/xstartup`
   - Starting applications specified in: `/home/vsadmin/.vnc/xstartup`
   - Log file is located in: `/home/vsadmin/.vnc/hqstage01.tibco.com:1.log`

# Third Party Libraries

In addition to the distribution provided by TIBCO, MDM requires additional software. This software must be provided for the installation and might have different licensing.

This table lists all the software which might be required.

*Third Party Libraries*

| Library | Library Name | Description | Vendor | How to obtain? |
|---|---|---|---|---|
| JDK Library | `jsse.jar` | Required, if you will be using SSL. | ORACLE/IBM/HP | Can be obtained from JDK. |
| XMLC related Libraries | `xmlc.jar`<br>`xmlc-base.jar`<br>`xmlc-chtml.jar`<br>`xmlc-taskdef.jar`<br>`xmlc-xerces.jar`<br>`xmlc-all-runtime.jar` | Required for TIBCO MDM UI. These components are supplied by Enhydra http://forge.ow2.org/projects/xmlc/. | | https://edelivery.tibco.com<br>Click the **XMLC Download** link under the TIBCO MDM area to download the `xmlc-2.2.x.zip`.<br>You can choose to download the library or let the installer download the library during installation process. |
| | `gnu-regexp.jar` | Required for compiling HTML. | | |
| EMS related Libraries | `tibjms.jar` | Required, if you are using TIBCO EMS as JMS vendor. | | The libraries can be obtained from installation directory of TIBCO EMS (pointed by EMS_HOME). |
| | Not applicable for the PostgreSQL database. | | | |

| Library | Library Name | Description | Vendor | How to obtain? |
|---|---|---|---|---|
| JDBC related Libraries | `ojdbc8.jar`<br>`mssql-jdbc-7.2.2.jre11.jar`<br>`postgresql-42.2.5.jar` | `ojdbc8.jar` is required for Oracle.<br>`mssql-jdbc-7.2.2.jre11.jar` ( Microsoft JDBC Driver 7.2 for SQL Server, Java Development Kit (JDK) 11.0 ) is required for Microsoft SQL Server.<br>`postgresql-42.2.5.jar` is required for PostgreSQL. (Not required if you are using Simple Installation) | | Copy the following JAR files to *$MQ_HOME*`/configurator/tomcat/ lib`:<br>• `ojdbc8.jar` copy from *$ORACLE_HOME*`/jdbc/lib`.<br>• Download the `mssql-jdbc-7.2.2.jre11.jar` ( Microsoft JDBC Driver 7.2 for SQL Server, Java Development Kit (JDK) 11.0 ) file from Microsoft Download Center.<br>• `postgresql-42.2.5.jar` file copy from *$MQ_HOME*`/bin/pgsql/driver` |
| Hibernate Assembly | `hibernate3.jar`<br>`cglib-2.2.jar` | Required by TIBCO MDM for some database interactions. | | https://edelivery.tibco.com<br>Click the **Download** link under the TIBCO MDM area to download the `product_tibco_hibernate_lgpl_3.6.10.003.zip`.<br>You can choose to download the library or let the installer download the library during installation process. |

All the required libraries are to be added to the distribution provided (ECM.ear) with TIBCO MDM.

## Creating a Consolidated JAR File with Third Party Libraries

For EMS related libraries, create a consolidated JAR file with the third party libraries. Other JAR files are created during installation.

Ensure the *$JAVA_HOME*`/bin` is set in the environmental variable *PATH*.

### Procedure

1. Create a third party folder in your local drive and copy all the third party libraries in the folder.
2. Go to the third party directory using the command prompt. For example: **`cd ../thirdparty`**
3. To create a JAR, for example, `ThirdParty.jar`, run the following command:

   ```
   $JAVA_HOME/bin/jar -cfM ThirdParty.jar *.jar
   ```

   This command creates a JAR file of all the `.jar` files located in the third party directory.

   After creating JAR files with the third party libraries, you need to merge the third party libraries with `ECM.ear` post TIBCO MDM installation. For information, refer to Performing Postinstallation Tasks for Typical Installation.

# Installation Checklist

Before you install MDM, you need to decide the various options and components.

*Installation Checklist*

| Check list | Description | Reference |
|---|---|---|
| Do you need to install more than one instance of MDM and cluster it? | More than one instance of MDM may be required to implement:<br>a) High availability.<br>b) support higher workloads.<br>When more than one instances is installed, you also need to install a web server to provide load balancing between the instances and to provide single URL for all instances. | Clustering Set Up<br>Configuration of Web Servers |
| Have you installed TIBCO EMS? | As part of the installation process, you need to configure TIBCO EMS for TIBCO MDM. | Configuration of Java Messaging Server |
| Do you plan to use text search? | Text searching provides support for fuzzy searches. | Text Search and Setup and Configuration sections in *TIBCO MDM System Administration* |
| Do you plan to customize cache servers? | Cache servers are required when large amount of data is to be cached and peer-to-peer topology is not sufficient. | • Apache Ignite<br>• TIBCO ActiveSpaces |
| Do you need GDSN support? | Global Data Synchronization plug in provides predefined models for synchronization with 1Sync. | • GDSN Overview<br>• Configuration of TIBCO BusinessConnect and TIBCO BusinessWorks |
| Do you need to enable UI for languages other than English? | MDM UI is localized and is supported in many languages. | Support for Language Pack |

- Plan the capacity. The capacity planning worksheet is available with TIBCO support which allows you estimate the required disk space, memory required for cache and CPU capacity required for processing workloads.
- Decide the components. For example, text indexing.
- Plan the deployment layout. Decide the components to be installed on different servers. Decide for sharing the common dir and configuration files. If you need more than one MDM instances and to know how such instances will share the common directory and configuration files, refer Clustering Set Up.
- Decide the Application server and database. As installation steps vary based on application server and databases, these are important decisions.

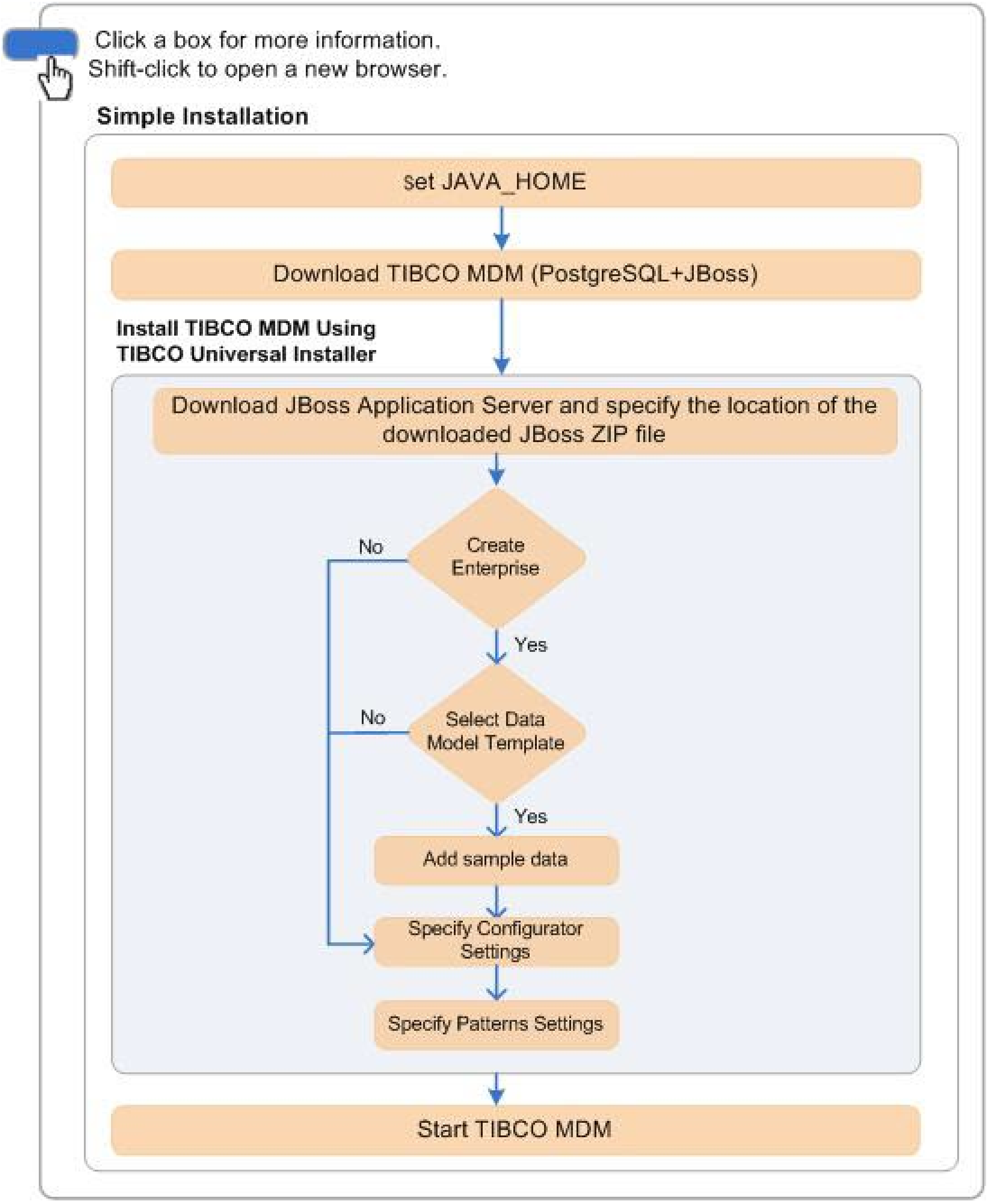
- Determine the languages in which you want the data to be stored in MDM. The storing of multiple languages requires the databases to be setup with correct options.
- Determine whether there are any custom components to be deployed with MDM.
- Decide whether you want to partition the data. You should consider partitioning the data if the data volume is high. For example, more than 300 M unique master data records. If the partitioning is required, contact TIBCO Support or Professional Services Group to understand how the partitioning can be done.

# Planning for Installation

TIBCO MDM provides two types of installations: simple and typical. Use the installation roadmaps to get started.

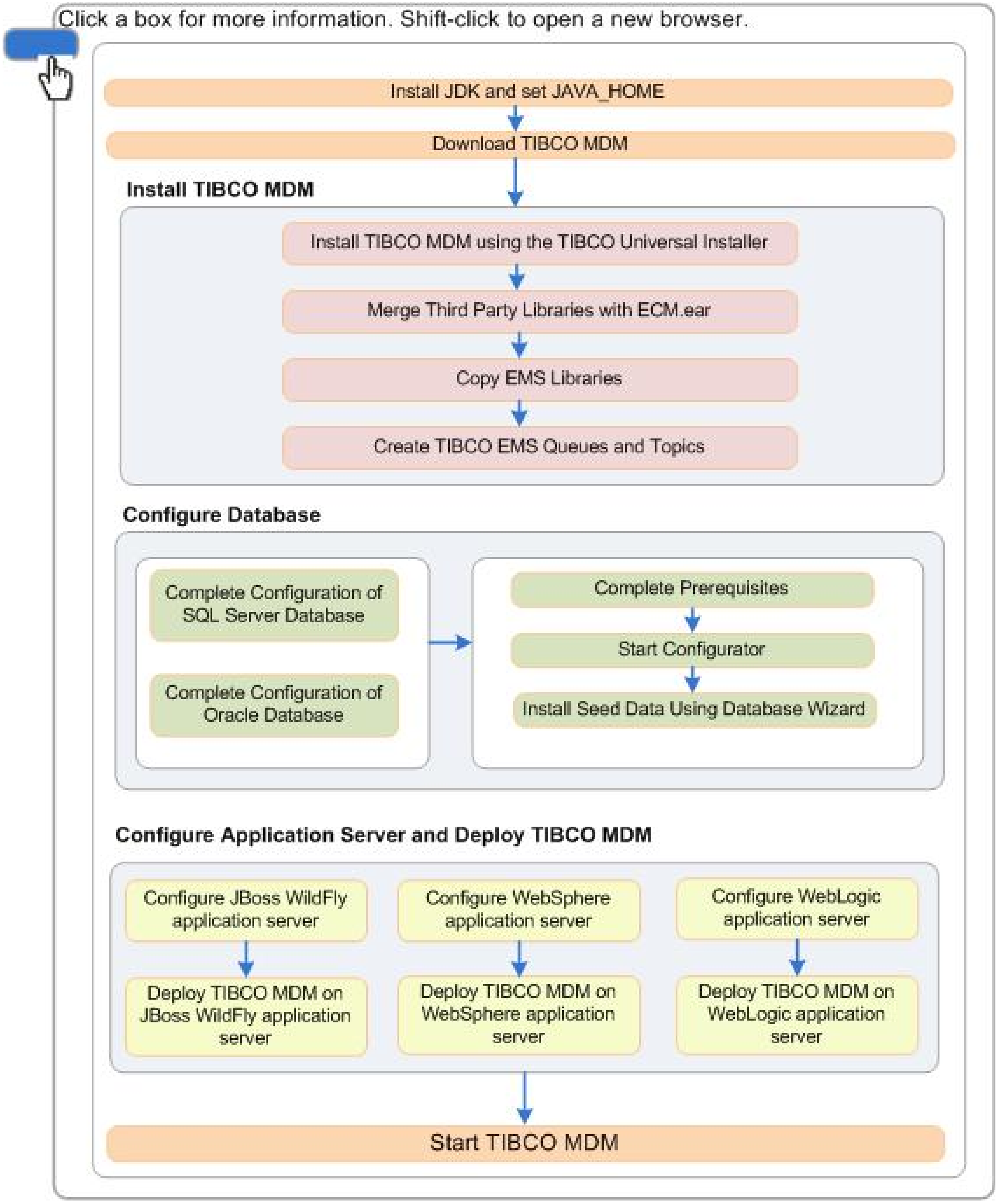
## Simple Installation Roadmap

The simple installation provides minimal prompts and installs standard components in the default locations. You can change the default locations, if required.


Click a box for more information.
Shift-click to open a new browser.
Simple Installation
set JAVA_HOME
Download TIBCO MDM (PostgreSQL+JBoss)
Install TIBCO MDM Using
TIBCO Universal Installer
Download JBoss Application Server and specify the location of the downloaded JBoss ZIP file
Create Enterprise
No
Yes
Select Data Model Template
No
Yes
Add sample data
Specify Configurator Settings
Specify Patterns Settings
Start TIBCO MDM

## Typical Installation Roadmap (Basic)

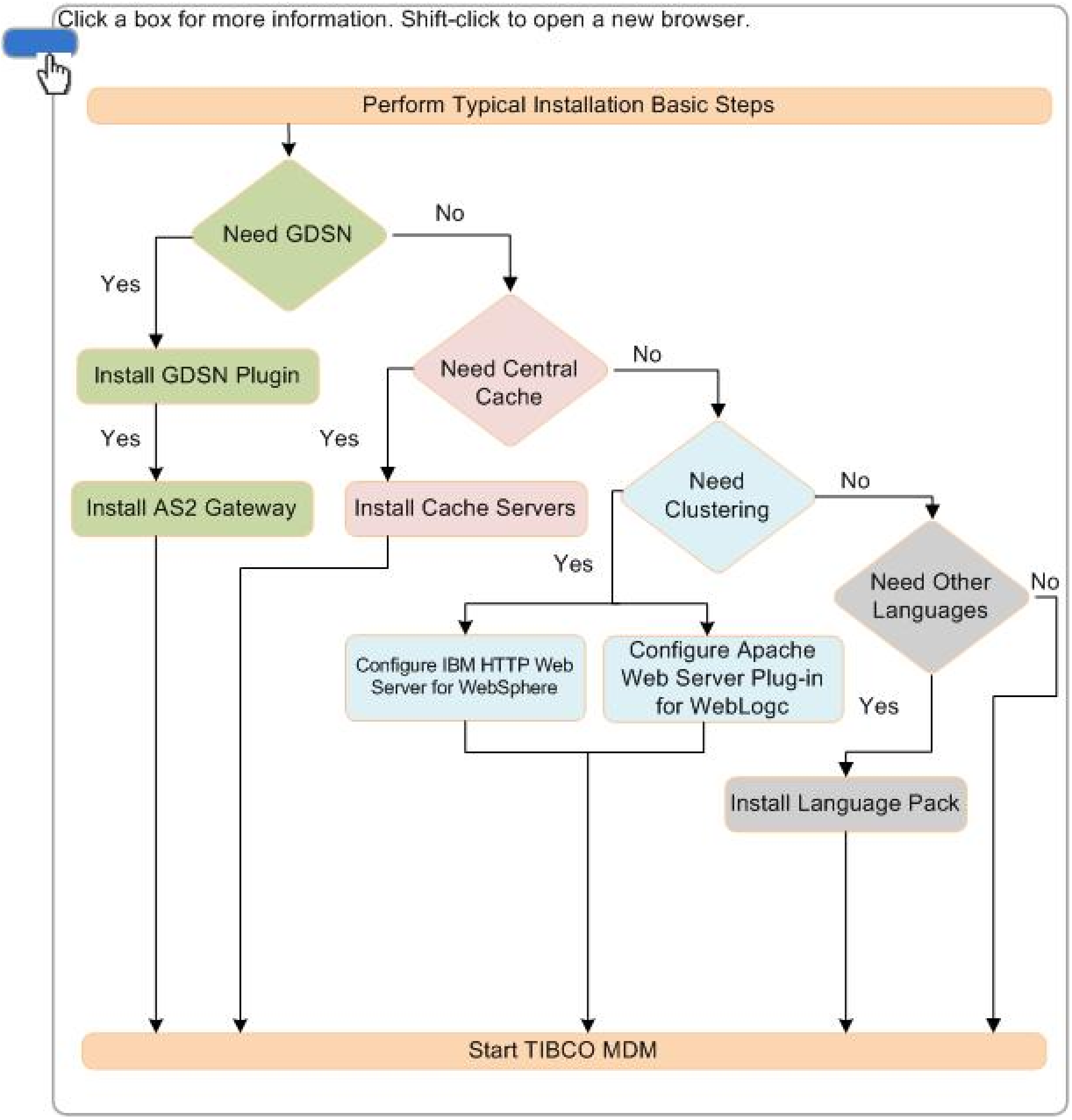
Prompts you to choose the basic components of the product that you want to install and installs only those components.



Click a box for more information. Shift-click to open a new browser.
Install JDK and set JAVA_HOME
Download TIBCO MDM
Install TIBCO MDM
Install TIBCO MDM using the TIBCO Universal Installer
Merge Third Party Libraries with ECM.ear
Copy EMS Libraries
Create TIBCO EMS Queues and Topics
Configure Database
Complete Configuration of SQL Server Database
Complete Configuration of Oracle Database
Complete Prerequisites
Start Configurator
Install Seed Data Using Database Wizard
Configure Application Server and Deploy TIBCO MDM
Configure JBoss WildFly application server
Deploy TIBCO MDM on JBoss WildFly application server
Configure WebSphere application server
Deploy TIBCO MDM on WebSphere application server
Configure WebLogic application server
Deploy TIBCO MDM on WebLogic application server
Start TIBCO MDM

## Typical Installation Roadmap (Advanced)

Prompts you to choose the advanced components of the product that you want to install and installs only those components.


Click a box for more information. Shift-click to open a new browser.
Perform Typical Installation Basic Steps
Need GDSN
No
Yes
Install GDSN Plugin
Need Central Cache
No
Yes
Yes
Install AS2 Gateway
Install Cache Servers
Need Clustering
No
Yes
Need Other Languages
No
Configure IBM HTTP Web Server for WebSphere
Configure Apache Web Server Plug-in for WebLogc
Yes
Install Language Pack
Start TIBCO MDM

# Configuration

The configurations are provided as a reference to plan the deployment. Based on the number of users and data volume, you can choose how to configure TIBCO MDM.

- Low-End Configuration: All components (web server, application server, JMS server, and database server) run on the same machine. This configuration is commonly used for development purposes.
- Mid-Range Configuration: The database server, application server, and web server run on separate machines. The JMS server can share the same hardware as the web server. This configuration is typically used for test environments.
- Mid-Range Clustered Configuration: This configuration is similar to mid-range configuration except that two or more application servers are clustered to support a large number of users. The clustered application supports load balancing using the round robin method. Optionally, a hardware or software based load balancer can be used to implement the required load balancing algorithm.
- High-End, High-Availability Configuration: This configuration supports high availability and high data volumes. The database, application server, and web server run on separate machines. More than one application server and web server are used for high availability. The database is paired with a standby machine. In this configuration, each component is paired for redundancy.

You can use a combination of low, mid, and high-end configurations.

## Configuration Based on Concurrent Users and Records

Use the following guidelines to determine which configuration is best for your company. Contact TIBCO Support for a detailed capacity planning spreadsheet.

*Recommended Configuration for Concurrent Users*

| Concurrent Users | Recommended Configuration |
|---|---|
| 1 to 20 | Low-end |
| 20 to 50 | Mid-range |
| 50 to 100 | Mid range clustered with 2 application servers. Each additional application server supports 30 additional concurrent users. |
| 100 to 500 | High-end |
| Over 1000 | Contact TIBCO Customer Support. |

*Recommended Configuration Based on Number of Records*

| Number of Records | Recommended Configuration |
|---|---|
| Less than one million | Low-end |
| 1 million to 10 million | Mid-range / Mid-range clustered |
| 10 million to 50 million | High-end |
| Over 50 million | Contact TIBCO Customer Support. |

## Hardware Configuration

The following tables lists sample hardware configurations. Additional memory may be required to accommodate data caching needs.

*Hardware Configurations*

| | Quantity | Type | | | Memory |
|---|---|---|---|---|---|
| Configuration | | Sun | IBM | Intel | |
| **Low End** | | | | | |
| Single machine for web server, application server, and DB server | 1 | Sun Fire X4100 - 2 CPU | IBM xSeries 2 CPU (AMD or Xeon) or equivalent | Xeon 2GHz, 2 CPU | 4 GB RAM, 50 -100 GB disk<br>8 GB RAM and Dual core processors are recommended to achieve increased throughput. |
| **Mid-range / Mid-range clustered** | | | | | |
| Web server | 1 | Sun Fire x2100 or Sun Fire x4100 equivalent, 1-2 CPU | IBM xSeries 1-2 CPU or pSeries entry level servers | Xeon 2 GHz, 1 CPU | 1 GB RAM, 36 GB internal disk |
| Application server | 1-2 | Sun Fire x4100 2 CPU with Dual core processors or Sun Fire V240 with 4 CPU | IBM i520 or IBM 630 with 2-4 core/CPU | Xeon 3 GHz, 2-4 CPU | 4- 6 GB RAM, 36 GB disk<br>8 GB RAM and dual core processors are recommended for higher throughput. |
| Database server | 1 | Sun Fire 445 or equivalent with 2-4 CPU | IBM i520 or P630 with 2-4 Core/CPU | Xeon 3 GHz, 2-4 CPU | 6-8 GB RAM, 200-500 GB disk. |
| Storage | 1 | | | | Disk array, disks of 100 - 200 GB. |
| **High-end** | | | | | |
| Web server | 1-2 | Sun Fire, V100, 1-2 CPU | IBM P610, 1-2 CPU | Xeon 2 GHz, 1-2 CPU | 2 GB RAM, 36 GB internal disks |

| | Quantity | Type | | | Memory |
|---|---|---|---|---|---|
| **Configuration** | | **Sun** | **IBM** | **Intel** | |
| Application server | 2-4 | Sun Fire V490 with 4 CPU or Sun Fire x4100 with 2 CPU dual core | IBM P650, 4-8 CPU | Xeon 3 GHz, 4-8 CPU | 6-8 GB RAM, 40 GB disk for each server<br>12 GB RAM recommended for higher throughput. |
| Database server | 1 | Sun Fire V4800,<br>4-8 CPU | IBM P650, 4-8 CPU | Xeon 3 GHz, 4-8 CPU | Minimum 12 GB RAM<br>400-600 GB disk. |
| Storage | 1 | | | | Disk array, disks of 200 to 500+ GBb. |

The hardware required depends on many factors including, number of concurrent users, usage patterns, retention of history and rate of change for the data. A more accurate capacity planning exercise should be done based on detailed scenario tests done in performance labs.

Contact TIBCO Professional Services or TIBCO Customer support for more details on how to calculate the required hardware. It is recommended that any production hardware planning be done using scenario based testing results. A sample capacity planning worksheet can be obtained from TIBCO Customer Support.

## Environment Variables

You must set the important environment variables before installing TIBCO MDM.

- It is recommended that you use ASCII characters for all file names. If these names include non-ASCII characters, copying the files from Windows to UNIX or Linux and vice versa may result in corruption of file names.
- While setting environment variables on all platforms, if the '\' character is used as a path separator instead of '/', it leads to errors as '\' is treated as an escape character.
- For Simple install, set only *JAVA_HOME* environment variable.
- For Typical install, set all the environment variables mentioned in the Environment Variables table.

*Environment Variables*

| Variable | Description |
|---|---|
| *MQ_HOME* | Define MQ_HOME to point to the installation directory. It is recommended that you allocate at least 8 GB to this directory. In a clustered environment, each application server should point to a separate location.<br>Example: /home/tibco/mdm/*version* |

| Variable | Description |
|---|---|
| *MQ_LOG* | The location where log files will be generated (the recommended location is $MQ_HOME/log). In a clustered environment, each server should point to a separate location. Define MQ_LOG to point to this directory. A minimum of 1 GB should be allocated to this directory. The best practice is to change the default location such that the directory is not a sub-directory of MQ_HOME.<br>Example: *$MQ_HOME*/log |
| *MQ_COMMON_DIR* | All standard configurations files for workflow and data validation as well as all customizations are stored in this directory. This directory also holds all files generated during normal application processing. It is shared by all application servers in the cluster, and should be mounted to each server. All disk space indicated in the section Hardware Configuration should be assigned to this directory, and the MQ_COMMON_DIR variable should be set. The best practice is to change the default location such that the directory is not a sub-directory of MQ_HOME.<br>Example: /home/tibco/mdm/*version*/common<br>If you plan to create a copy of the TIBCO MDM instance across operating systems (for instance, Linux to Windows or Windows to Linux) and if the path contains any non English characters, such a copy may not be possible. For example, using Japanese characters in the path. |
| *MQ_CONFIG_FILE* | Points to *$MQ_HOME*/config/ConfigValues.xml. The values/parameters in this file can be set using the Configurator.<br>Example: /home/tibco/mdm/*version*/config/ConfigValues.xml |
| *JAVA_HOME* | The directory where JRE/JDK is installed.<br>Example: /opt/jdk*version* |
| *EMS_HOME* | The directory where TIBCO EMS (or the messaging software) is installed.<br>Example: /home/tibco/ems |
| *ANT_HOME* | The Directory path where ant is installed.<br>Example: /opt/ant*version* |
| **Application Server Specific** | |
| *WAS_HOME* | The directory where WebSphere is installed (required *only* if using WebSphere).<br>Example: /opt/WebSphere/AppServer |

| Variable | Description |
|---|---|
| *JBOSS_HOME* | For JBoss WildFly Application Server. Specify the path value until the root of the WildFly directory.<br>Example, `E:\JBoss\wildfly-`*`version`*`.Final`. |
| *JBOSS_HOME* `(Simple Install)` | The directory where JBOSS is installed (required only if using simple installation).<br>Example: *%MQ_HOME%*`\bin\wildfly-`*`version`*`.Final` |
| *BEA_HOME* | For WebLogic Application Server. Specify the path value of the WebLogic Application Server directory.<br>Example:<br>• For Linux: `/opt/bea`<br>• For Windows: `G:/WebLogic/wlserver_`*`version`* |
| **Database Specific** | |
| *ORACLE_HOME* | For Oracle database. The directory where Oracle is installed.<br>Example:<br>• For Windows: `/home/oracle/product/11g/db_1`<br>• On UNIX: `$export` *`ORACLE_HOME`*`=/u01/app/oracle/product/`*version* |
| *LD_LIBRARY_PATH* | For Oracle database: *$ORACLE_HOME*`/lib` |
| NLS_LANG | For Oracle database. Example:<br>• On UNIX:<br>`export NLS_LANG=AMERICAN_AMERICA.UTF8`<br>• On Windows:<br>`set NLS_LANG=AMERICAN_AMERICA.UTF8` |
| *POSTGRESQL_HOME* | For PostgreSQL database. The directory where PostgreSQL is installed.<br>Example: *$MQ_HOME*`/bin/pgsql` |
| *AS_HOME* | The directory where AS is located. By default AS is bundled with TIBCO MDM, if you have installed AS externally the specify the path of the location.<br>Example:`$MQ_HOME/bin/as/`*`version`* |
| *OS* | The Operating system. For example, Linux. |

| Variable | Description |
|---|---|
| *DISPLAY* | This environment variable is used by X-Windows based applications. It points to a device capable of displaying an X-Windows based UI. |
| *LD_ASSUME_KERNEL* | Used on the Linux platform to make Linux use the old Linux threads library, particularly required for Oracle installation (required *only* if Oracle is used as the database). |
| *PATH* | This is a list of directories separated by a separator. When any command or program is executed, the OS tries to locate the program in the directories listed in PATH. If the program is not found in any of the directories, the OS cannot load and execute the program. The Separator character is ':' for Unix and Linux platforms, and ';' for the Windows platform.<br>Ensure that there is no space with the commas and colons between the program in the directories listed which are separated by ':' for Unix and Linux platforms, and ';' for the Windows platform. |
| *SHLIB_PATH* | List of directories separated by a separator (see `PATH`) where a dynamic linker tries to find the libraries. Used on UNIX platforms. |
| *LIBPATH* | List of directories separated by a separator (see `PATH`) where the Operating system as well as the application library files reside. Used on UNIX platforms. |
| *NODE_ID* | Points to the current cluster member. Example:<br>`NODE_ID=Member1` |
| **TIBCO MDM REST API through Swagger UI** | |
| *SWAGGER_MDM_HOST* | An IP address of the TIBCO MDM server on which you want to try out the TIBCO MDM REST APIs. |
| *SWAGGER_MDM_PORT* | Port of the TIBCO MDM server on which you want to try out the TIBCO MDM REST APIs. |

## Storage Requirements

The space allocation requirements depend on many factors, including the number of records and the number of messages sent and received from other applications.

Space is required for the following components:

- Common directory
  - Work
  - Temp
- Database

- Log directory
- Local disk for Application Server
- Installation directory
- Other Software

For most development and test environments, 10 GB is sufficient. However, the disk space required for production and most user acceptance test environments is much higher. A sample capacity planning worksheet can be obtained from TIBCO Customer Support, which can help you calculate the disk capacity required.

As disks starts to fill up, you can archive or purge data. Space allocation varies according to the storage systems used and depending on how the disks are arranged, including any mirroring and archive log retention policies.

***Storage Requirement Components***

| Component Name | Description |
|---|---|
| Common Directory | TIBCO MDM stores some configuration and temporary files on disks. It also stores files associated with master data on disks.<br>TIBCO MDM stores certain data in files, including data imported and received as messages and data output from the application. This type of data is stored on file systems, with an entry in the database.<br>Disk requirements depend on size of the files (which directly relate to number of attributes and size of data for each record), retention period, and rate of changes to data. Additionally, the frequency of data synchronization with other systems and number of such systems may also influence the disk space requirements. A sample capacity planning worksheet can be obtained from Customer Support. File systems has work and temp directories besides other smaller directories. |
| Database | TIBCO MDM stores most of the master data in a database. The storage needs of the database depend on various factors including rate of change, data retention policies and complexity of the data model.<br>The disk space requirements vary a lot based on usage patterns and the following numbers should be used as indicative only:<br>Small: 5GB<br>Medium: 10-50GB<br>Large: 50-500GB |

| Component Name | Description |
|---|---|
| Log Directory | It is recommended that 1 GB space be allocated for the log directory so that sufficient number of debug logs can be accumulated. The debug logs are generated when TIBCO MDM runs in the debug mode. Using the Configurator, you can configure the debugging parameters.<br>You can select the `DEBUG` option for the Debug Log Logging Level Threshold parameter. The other options are `FATAL`, `ERROR`, `WARN`, and `INFO`. The size of the logs is determined by the following two parameters:<br>• Debug Log File Backup Size: Indicates the number of debugging log backup files. By default, the size is `70`.<br>• Debug Log Maximum File Size- Indicates the maximum size of the debugging log file. By default, the log file size is `5MB`.<br>The log file location is specified by the Standard Log Appender File parameter. By default, the location is *`$MQ_LOG`*/elink.log. |
| Local Disk for Application Server | The application server needs at least 6 GB of local free disk space to deploy the application. This storage is separate from `$MQ_COMMON_DIR (8 GB)`and *`$MQ_HOME`* (8 GB), and is used by the application server. |
| Installation Directory | This is where MDM is installed. 700 MB is recommended. |
| Other Software | Additional disk space is required for:<br>• JMS server<br>• Web server<br>• Backups, if taken on disks<br>Refer to the appropriate documentation from the software vendor for an estimate of the space required. |

## Impact of In-Memory Workflows

Workflow can be defined to run in-memory. Such workflows produce less persistent data and hence require less disk space, both on the file system and the database.

For more information on running workflows in-memory, refer to *TIBCO MDM Workflow Reference*.

### Failover considerations

It is recommended that only smaller workflows be run in-memory and not long running workflows, since in case of failover for an in-memory workflow, the entire workflow is re-executed. For more information on workflow failover, refer to *TIBCO MDM Workflow Reference*.

## Connectivity with External Systems

Skip this section if you do not plan to perform Global Data Synchronization (GDSN).

To connect with 1Sync or similar other data pools, or with any other system requiring AS2 connectivity, you need to install and configure one of the supported AS2 servers as gateway.

If you want to setup connectivity with your backend systems using AS2, you need to install and configure a compatible AS2 software. Alternatively, you can use a JMS based communication method.

## Internationalization

To support internationalization (I18N) on your system, follow the internationalization guidelines.

*Internationalization Guidelines*

| Component | Description |
|---|---|
| Operating System Layer | Verify that the locale is set correctly to match the character set you will be using, or to UTF8. Consult your OS documentation for information on how to do this. |
| Application Server Layer | Verify that the JVM arguments for encoding - file encoding and client encoding (file.encoding, client.encoding.override) - are both set to UTF-8.<br>This setting is recommended irrespective of whether you want to support internationalization. |
| Oracle Database: | To ensure that language support or multi-byte support is consistent throughout the Oracle instance, set the language support at the time of setting up an Oracle instance. To support multiple languages, set the flag to UTF-8. Refer to Configuration of Oracle Database for instructions. |
| PostgreSQL Database | To ensure that language support or multi-byte support is consistent throughout the PostgreSQL instance, set the language support at the time of setting up an PostgreSQL instance. To support multiple languages, set the Encoding to UTF-8. |
| SQL Server Database | To ensure that language support or multi-byte support is consistent throughout the SQL Server instance, set the language support at the time of setting up an SQL Server instance. To support multiple languages, set the Collation flag to the native language with Unicode, for example: Japanese_Unicode_CI_AS. |
| Internet Explorer | If characters in certain language are not displayed correctly in the browser, on the **View** menu of Internet Explorer, point to **Encoding**, point to **More**, and then click the appropriate language.<br>If the specific language pack is not installed on the computer, you will be prompted to download language support components; click **Download**. |
| Google Chrome | If characters in certain language are not displayed correctly in the browser, on the **View** menu of Chrome, point to **Encoding**, and then click the appropriate language. |
| Fire Fox | If characters in certain language are not displayed correctly in the browser, on the **View** menu of Fire fox, point to **Character Encoding**, and then click the appropriate language or point to **More Encodings** to find more languages option. |
| Data Files and XML | If you want to upload data using data sources, ensure that data files are saved with UTF-8 encoding for text files. For any XML file upload, verify that the encoding is set as follows:<br>“<?xml version=’1.0’ encoding=’UTF-8’?>” |

## Single Sign-Ons

You can set up TIBCO MDM to be used along with other authentication servers to support single sign-on.

Single sign-on with other authentication servers is not mandatory. If you do not use an authentication server for single sign-on or do not plan to manage user authentication for TIBCO MDM logon using authentication servers, skip this section.

The single sign-on plugins are provided as examples. These can be customized to implement other methods of single sign-on. The supported platforms are:

- Any LDAP v3 compliant server, such as, Sunone Directory Server.
- IBM's Tivoli Access Manager.
- Computer Associates eTrust SiteMinder.

*Single Sign-Ons*

| Types of Single Sign-on | Description |
|---|---|
| LDAP Single Sign-On | With LDAP, you are expected to use LDAP authentication to access the TIBCO MDM application and the same LDAP information can be used to bypass the first login screen if single sign-on is configured (for instance, SiteMinder or TAM or Oblix).<br>**Preferred LDAP Platforms**<br>Verify that the systems and software at your site meet the following requirements:<br>• Sunone Directory Server<br>• LDAP Server: Oblix COREid<br>• Tivoli Access Manager (TAM) |
| SiteMinder Single Sign-on | Organizations using Computer Associates eTrust SiteMinder for access management can configure TIBCO MDM to work with SiteMinder. When TIBCO MDM is configured to use the user authentication from SiteMinder, end users will bypass the TIBCO MDM login screen when accessing the TIBCO MDM application. |

## Clustered Deployment Recommendations

If you are deploying TIBCO MDM in a cluster (more than one application server), you can deploy with one of these options.

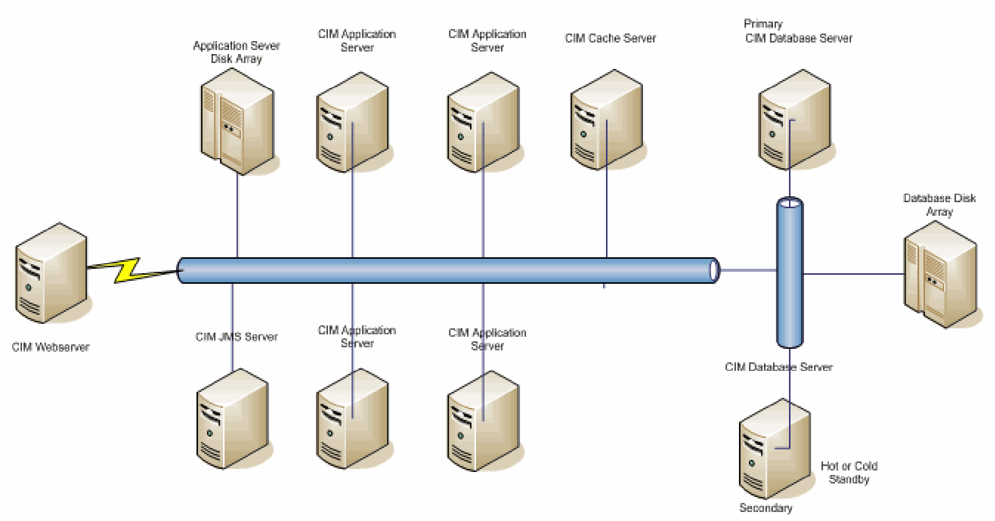
*Clustered Deployment Recommendations*

| Options | Description |
|---|---|
| Option 1 (recommended) | `MQ_HOME` is on a common file system which is shared between all application servers. The `ConfigValues.xml` file has configuration for all instances. The advantage is that servers are guaranteed to run same version. Any mismatch of version will result of failures. In both options, `MQ_LOG` is not shared, each application server should have its separate log directory. It is recommended that `MQ_LOG` points to internal disks to reduce the disk write overhead. `MQ_COMMOM_DIR` is on a common file system which is shared between all application servers. |
| Option 2 | Each application server has a separate `MQ_HOME` where all application images are stored. The `config/` directory must be identical for all application servers, including `ConfigValues.xml`. The `ConfigValues.xml`is to be managed using the Configurator.<br>This option should be used only if *`MQ_HOME`* sharing is not possible. Care should be taken to ensure that the `config/` directory is synchronized between all instances whenever a configuration change is made.<br>`MQ_COMMOM_DIR` is on a common file system which is shared between all application servers. |

## Typical Deployment

A typical production deployment with high availability is illustrated in the Typical Deployment figure.

*Typical Deployment*



# Configuration of Java Messaging Server

As part of the installation process, you can configure TIBCO EMS for TIBCO MDM. In general, this involves creating queues and configuring associated property files.

### Predefined Queues and Topics

TIBCO MDM provides some predefined queues and topics. For a complete list of queues and topics, refer the "Queue Management" chapter in *TIBCO MDM System Administration*.

You can customize queue and topic names and also map logical and physical queue and topic names using Configurator.

> If TIBCO MDM changes the format of these configuration files in future, all customizations made to these files may need to be applied to new files shipped with the new version of TIBCO MDM.

## Creating TIBCO EMS Queues and Topics

The required queues and topics are located in the `$MQ_HOME/bin/install/createQueues.txt` file.

### Prerequisites

Ensure that the EMS Server is running.

### Procedure

1. Go to **All Programs** > **TIBCO** > **TIBCO EMS** ***version*** and click **Start EMS Administration Tool**. The command prompt is displayed.
2. Type **Connect**.
3. Enter login name and password.

   > If you have not created the administration credentials, press **Enter**. By default, administration login name and password is retrieved.

   The connected to: `tcp://localhost:`*port_number* message is displayed.
4. Go to `$MQ_HOME/bin/install` and open the `createQueues.txt` file.
5. Copy content of the `createQueues.txt` file and place it in the command prompt. Queues and topics are created.

## Verifying Queues and Topics

You can verify a list created queues and topics.

### Procedure

1. Go to Go to **All Programs** > **TIBCO** > **TIBCO EMS** ***version*** and click **Start EMS Administration Tool**. The command prompt is displayed.
2. Type **Connect**.
3. Enter login name and password.
4. Type show queues in the command prompt and press **Enter**. A list of created queues is displayed.
5. Type show topics in the command prompt and press **Enter**. A list of created topics is displayed.

> For TIBCO MDM 8.x versions with EMS 6.x, while creating queues and topics using the `$MQ_HOME/bin/createQueues.txt` script; Queues can be created using the command create queue Q_ECM_CORE_WORKFLOW store=$sys.failsafe Topics can be created using the following command create topic T_ECM_CORE_ADMIN store=$sys.failsafe

## Configuring EMS over SSL

You can configure the EMS server to start running over SSL.

**Prerequisite**

Ensure that `tibcrypt.jar` is in the CLASSPATH of the application server.

### Procedure

1. Stop the application server.
2. Edit the following values in *$EMS_HOME*`\tibco\cfgmgmt\ems\data\tibemsd.conf` file.
    - specify the SSL protocol in the listen parameter: `listen= ssl://`*`hostname`*`:`*`portno`*
    - `ssl_server_identity = `*`TIBCO_HOME`*`/ems/`*`version_number`*`/samples/certs/server.cert.pem`
    - `ssl_server_key = `*`TIBCO_HOME`*`/ems/`*`version_number`*`/samples/certs/server.key.pem`
    - `ssl_password = $man$WjtSRCpaXu7hoTkDlcEPr6KNKRr`
    - `ssl_server_trusted =`*`TIBCO_HOME`*`/ems/version_number/samples/certs/client_root.cert.pem`
3. Start EMS server using the updated `tibemsd.conf` file.

    *`TIBCO_HOME`*`/ems/`*`version_number`*`/bin/tibemsd -config `*EMS_HOME*`/tibco/cfgmgmt/ems/data/tibemsd.conf`

    The EMS server starts running over SSL.

## Configuring EMS over SSL on Application Servers

To configure EMS over SSL, configure the application servers and then specify the values for the TIBCO EMS configuration properties in Configurator.

### Procedure

1. Configuring WebSphere Application Server
    a) Stop the application server.
    b) Copy the following JAR files from `$EMS_HOME/lib` to `$WAS_HOME/lib/ext` folder:
    - `slf4j-api-1.4.2.jar`
    - `slf4j-simple-1.4.2.jar`
    - `tibcrypt.jar`
2. Configuring JBoss WildFly Application Server
    a) Create a module with name as `com.tibco.mdm` in JBoss WildFly application server - 10.1.x version. For more information on creating module, see Module Creation.
    b) Copy the following JAR files in this module (inside main directory) from `$EMS_HOME/lib`
    - `slf4j-api-1.4.2.jar`
    - `slf4j-simple-1.4.2.jar`

- tibcrypt.jar

c) Copy the following JAR files in this module. The JAR files are available in the classpath of the JBoss WildFly application server or in `JAVA_HOME\jre\lib` directory:

- jsse.jar
- jce.jar

d) Update the `module.xml` file as follows:

```
<module xmlns="urn:jboss:module:1.3" name="com.tibco.mdm">
  <resources>
    <resource-root path="tibcrypt.jar"/>
    <resource-root path="slf4j-api-1.4.2.jar"/>
    <resource-root path="slf4j-simple-1.4.2.jar"/>
    <!--  if these 2 JARS copied in the module then add -->
    <resource-root path="jsse.jar"/>
    <resource-root path="jce.jar"/>
  </resources>
<dependencies>
   <module name="javax.api" />
   <module name="javax.jms.api" />
   <module name="javax.resource.api" />
   <!-- These are required for EMS with SSL -->
   <system export="true">
      <paths>
         <path name="sun/security/ssl" />
         <path name="com/sun/net/ssl/internal/ssl" />
         <path name="sun/security/util" />
         <path name="sun/security/validator" />
         <path name="sun/security/provider" />
         <path name="javax/net/ssl" />
         <path name="sun/net/www/protocol/https" />
      </paths>
   </system>
</dependencies>
```

e) Add the global module in subsystem section `<subsystem xmlns="urn:jboss:domain:ee:4.0">` in *JBOSS_HOME*`/standalone/configuration/standalone.xml`.

```
<subsystem xmlns="urn:jboss:domain:ee:4.0">
  <global-modules>
    <module name="com.tibco.mdm" slot="main"/>
  </global-modules>
</subsystem>
```

OR

Add a dependency in `jboss-deployment-structure.xml` in `ECM.ear/META-INF` in dependency section:

```
<dependencies>
  <system export="true">
    <paths>
      <path name="com/sun/net/ssl/internal/ssl" />
    </paths>
  </system>
  <module name="com.tibco.mdm"/>
</dependencies>
```

f) Add the following in the JVM arguments in *$JBOSS_HOME*`/bin/standalone.conf.bat`

```
Dcom.tibco.tibjms.ssl.enable_verify_host_name=false
Dcom.tibco.tibjms.ssl.enable_verify_host=true
Dcom.tibco.tibjms.ssl.expected_hostname=server
Dcom.tibco.tibjms.ssl.trusted=$EMS_HOME/samples/certs/server_
root.cert.pem
```

3. Configuring TIBCO MDM

a) Log on to Configurator.

b) Navigate to **Queue Setup** > **Messaging Cluster** > **TIBCO EMS** .

Change the value of Localhost Server Connection String property to **ssl://***hostname***:***portno*

c) Navigate to **Bus Setup** > **Cluster** > **TIBCO EMS** .

   Change the value of Localhost Server Connection String property to **ssl://***hostname***:***portno*

d) Navigate to **InitialConfig** > **Member1** > **Security Provider** > **IBM**

   - change the value of SSL Protocol Handler Package property from: `com.ibm.net.ssl.internal.www.protocol`to `com.ibm.net.ssl.www2.protocol`
   - change the value of SSL Provider property from: `com.ibm.jsse.JSSEProvider` to `com.ibm.jsse2.IBMJSSEProvider2`

e) Add the following in the JVM arguments:

```
Dcom.tibco.tibjms.naming.security_protocol=ssl
Djsse.providerClass=com.ibm.jsse2.IBMJSSEProvider2
Dcom.tibco.tibjms.ssl.expected_hostname=server
Dcom.tibco.tibjms.ssl.enable_verify_host_name=false
Dcom.tibco.tibjms.ssl.enable_verify_host=false
Dcom.tibco.tibjms.ssl.trusted=/local/vsadmin/server_root.cert.pem (you can
copy this out-of-box certificate from $EMS_HOME/samples/certs folder)
```

4. Start the application server.
5. Log on to TIBCO MDM.

   TIBCO MDM starts running on SSL.

# TIBCO MDM Installation

You can run the TIBCO installer in different modes, supported on all platforms.

TIBCO MDM may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. Refer to the *Readme.txt* file to locate whether or not TIBCO MDM is available on a particular operating system.

If you select *Simple Installation with PostgreSQL*, you can skip the sections from TIBCO MDM Installation on Application Servers to Support for Language Pack.

## Simple Installation with PostgreSQL

The simple installation provides minimal prompts and installs standard components in the default locations. You can change the default locations, if required.

The simple installation includes the following features:

- Currently supports JBoss WildFly Application Server and PostgreSQL database on Windows and Linux.
- Similar to the existing installer, in addition, it bundles the following dependencies except EMS.
  - ActiveSpaces
  - TIBCO Patterns (Netrics)
  - PostgreSQL Database
  - JBoss modules
- Installs dependencies in `$MQ_HOME/bin` directory. Therefore, not much control over the directories.
- Provides an additional screen to download the JBoss WildFly Application Server, that is, the installer zip file.
- Provides default settings, thus TIBCO MDM is configured automatically.

### Installing TIBCO MDM (Simple Installation)

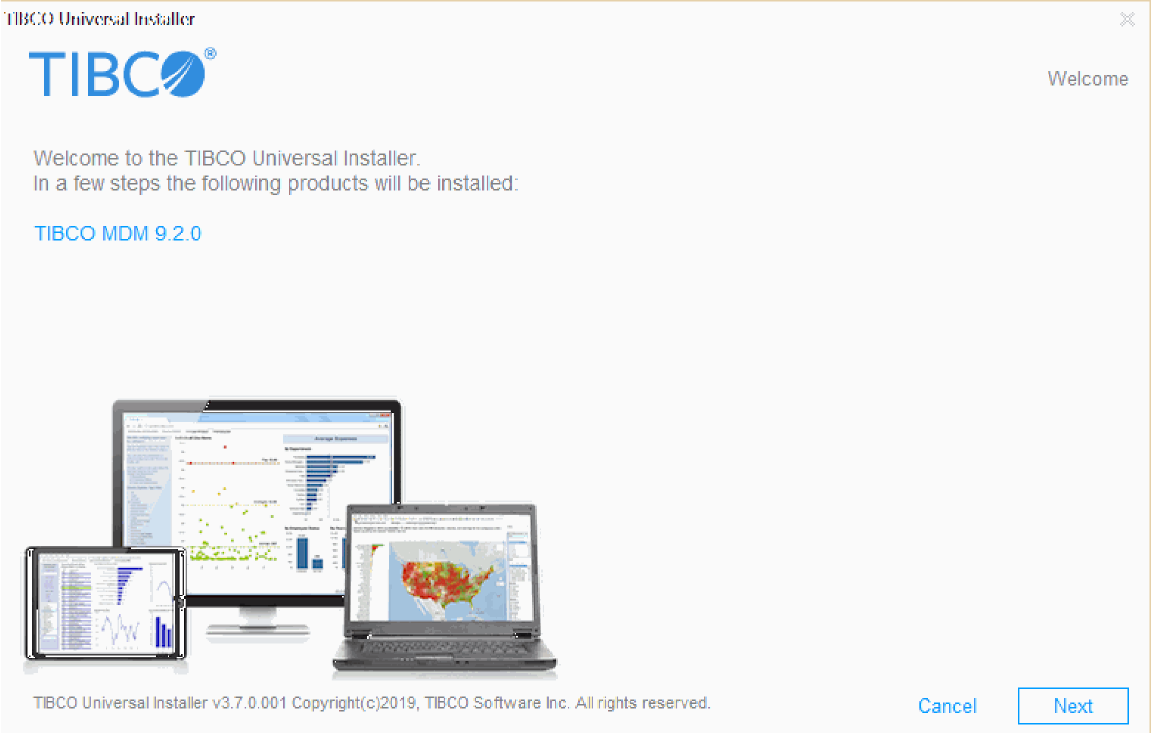
In Simple installation, the installer bundles, installs, and configures most of the required softwares. You need not worry about dependencies. Thus, it simplifies the installation process.

**Prerequisites**

- Verify that your computer meets the system requirements to install TIBCO MDM for PostgreSQL. The system requirements are listed in the `readme.txt` file. All the software components are installed under `$MQ_HOME`. Ensure that you have at least 5 GB of disk space.
- Set the `$JAVA_HOME` environment variable.
- Optional: Download the `wildfly-16.0.0.Final.zip` file from `http://wildfly.org/downloads/`
- If you are installing in a Linux environment, ensure that you are a non ROOT user.
- TIBCO MDM does not use the existing PostgreSQL instance. It re-installs and re-configures PostgreSQL. Ensure that you have stopped the current PostgreSQL instance, if running. Ensure that the default port `5432` is not in use.
- Download TIBCO MDM from https://edelivery.tibco.com. The application is available as a ZIP file.
- Extract contents of the ZIP file to a folder on your computer.
- Search for the TIBCOUniversalInstaller application in the location where you have extracted the ZIP file.

### Procedure

1. Run the `TIBCOUniversalInstaller.exe` application. The TIBCO Universal Installer Welcome window is displayed.



TIBCO Universal Installer

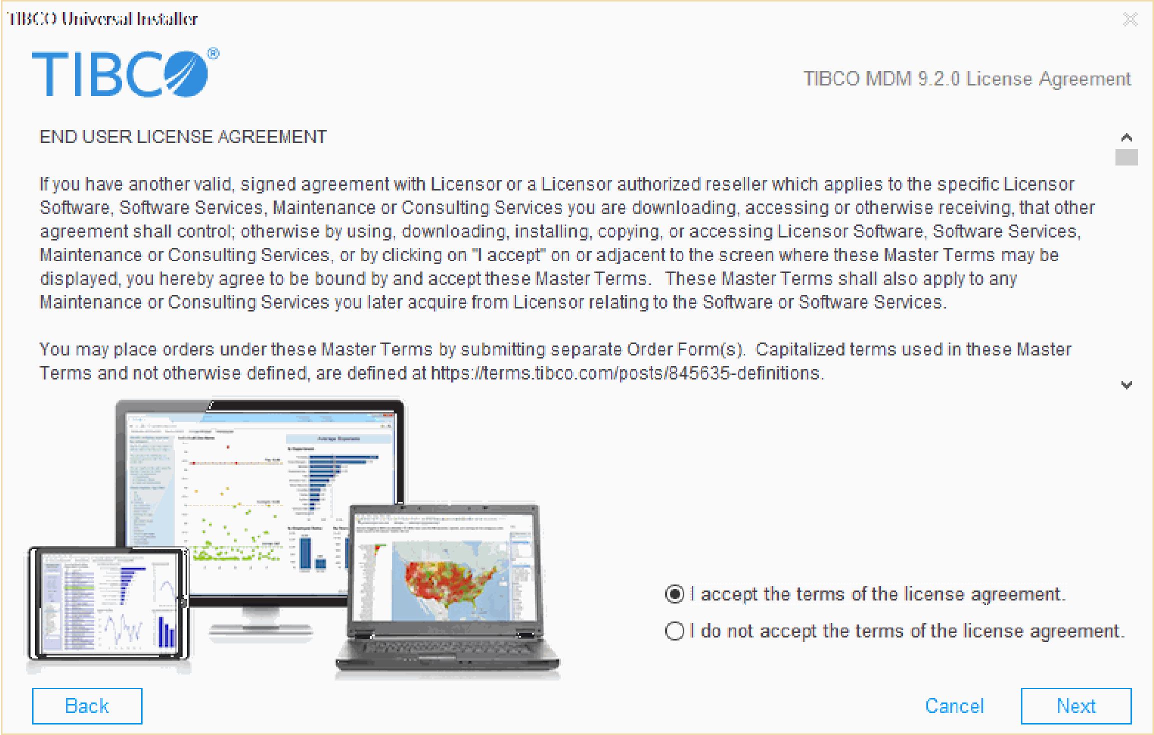
Welcome

Welcome to the TIBCO Universal Installer.
In a few steps the following products will be installed:

TIBCO MDM 9.2.0

TIBCO Universal Installer v3.7.0.001 Copyright(c)2019, TIBCO Software Inc. All rights reserved.

2. Review the information and click **Next**. The license agreement is displayed.



TIBCO Universal Installer

TIBCO MDM 9.2.0 License Agreement

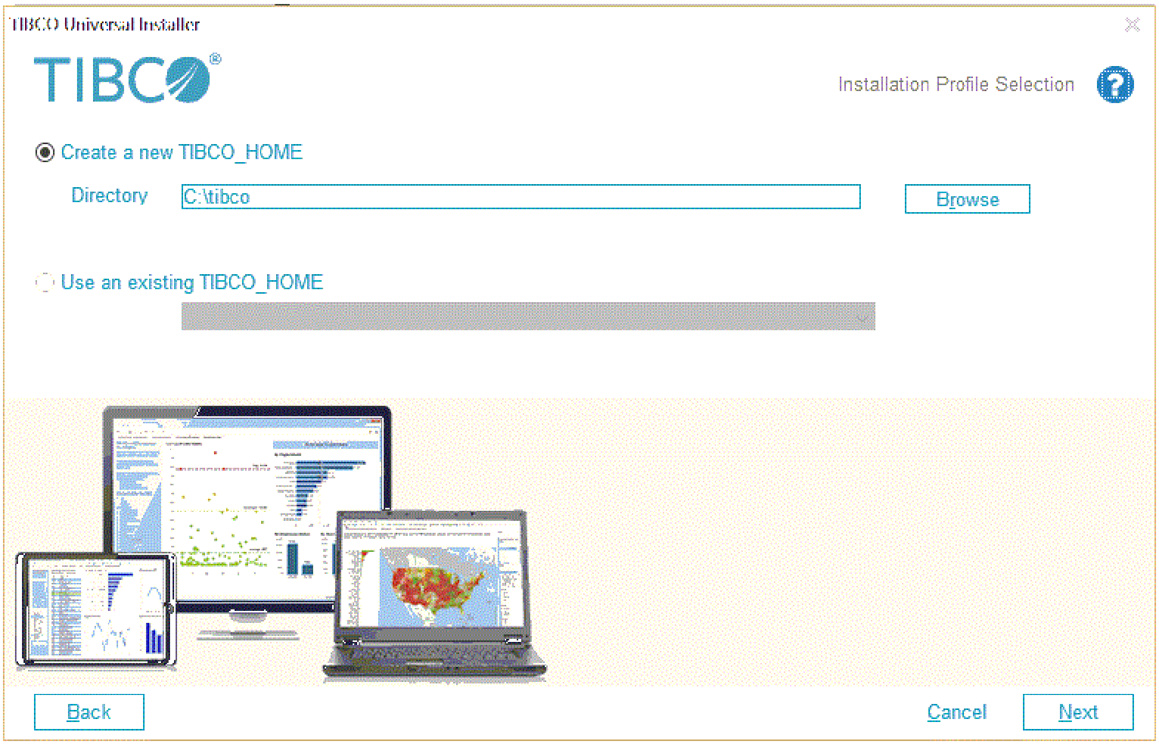
END USER LICENSE AGREEMENT

If you have another valid, signed agreement with Licensor or a Licensor authorized reseller which applies to the specific Licensor Software, Software Services, Maintenance or Consulting Services you are downloading, accessing or otherwise receiving, that other agreement shall control; otherwise by using, downloading, installing, copying, or accessing Licensor Software, Software Services, Maintenance or Consulting Services, or by clicking on "I accept" on or adjacent to the screen where these Master Terms may be displayed, you hereby agree to be bound by and accept these Master Terms. These Master Terms shall also apply to any Maintenance or Consulting Services you later acquire from Licensor relating to the Software or Software Services.

You may place orders under these Master Terms by submitting separate Order Form(s). Capitalized terms used in these Master Terms and not otherwise defined, are defined at https://terms.tibco.com/posts/845635-definitions.

● I accept the terms of the license agreement.
○ I do not accept the terms of the license agreement.

3. Review the terms of the license agreement. If you agree with its terms, accept the license agreement and click **Next**.
4. In the Installation Profile Selection window, do one of the following:



● Create a new TIBCO_HOME

Directory: C:\tibco

○ Use an existing TIBCO_HOME

a) Select **Create a new TIBCO_HOME** option. A TIBCO installation environment is used for software installations and consists of a Name and Directory. Products installed into different installation environments do not share components; therefore you can keep product installations completely isolated from each other.

- **Directory**: Browse to the directory where product needs to be installed and provide a unique environment name. Ensure that you have write permission to this directory.

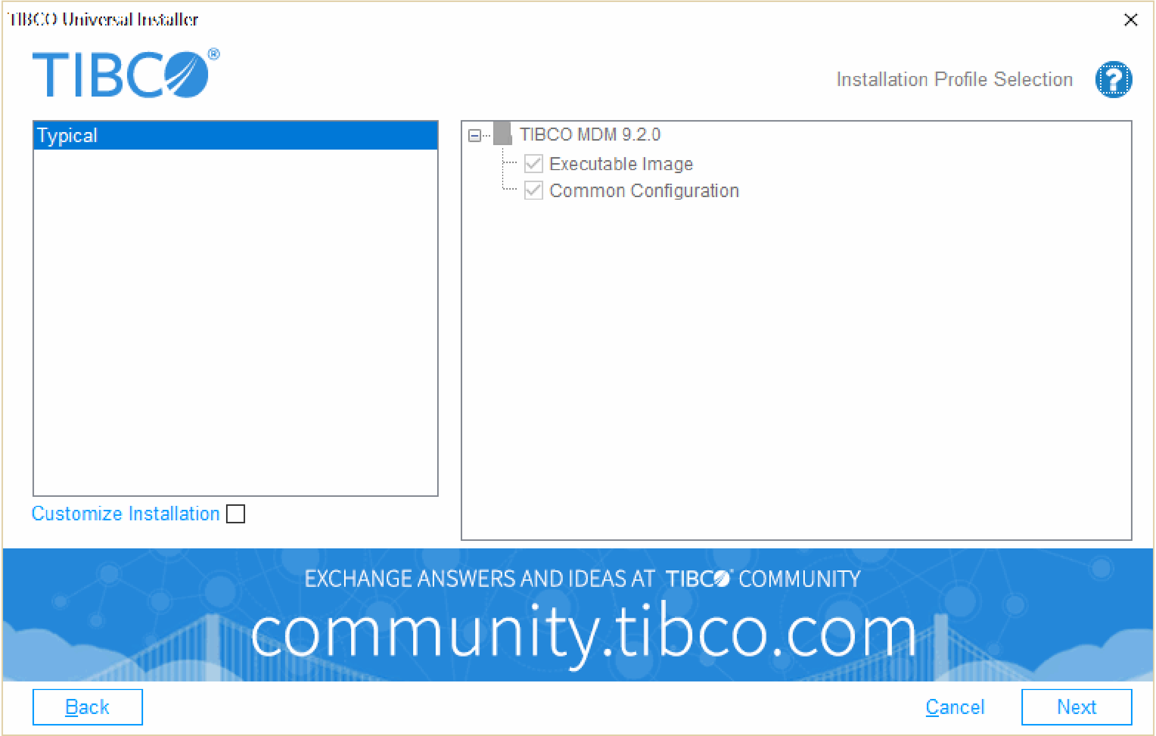
  On Microsoft Windows, it is recommended that you do not install under `C:\Program Files` or any other directory which contains spaces in the name. The Simple installer does not recognize the path if the folder name contains a space.

- **Name**: Specify the environment name.

b) If you have previously installed a TIBCO product using the Universal Installer, you can select **Use an existing TIBCO_HOME**. By default, the installer detects the directory for your TIBCO_HOME and displays the path.

   If you had installed the application earlier, the directory cannot be modified. For a new installation, the directory can be changed.

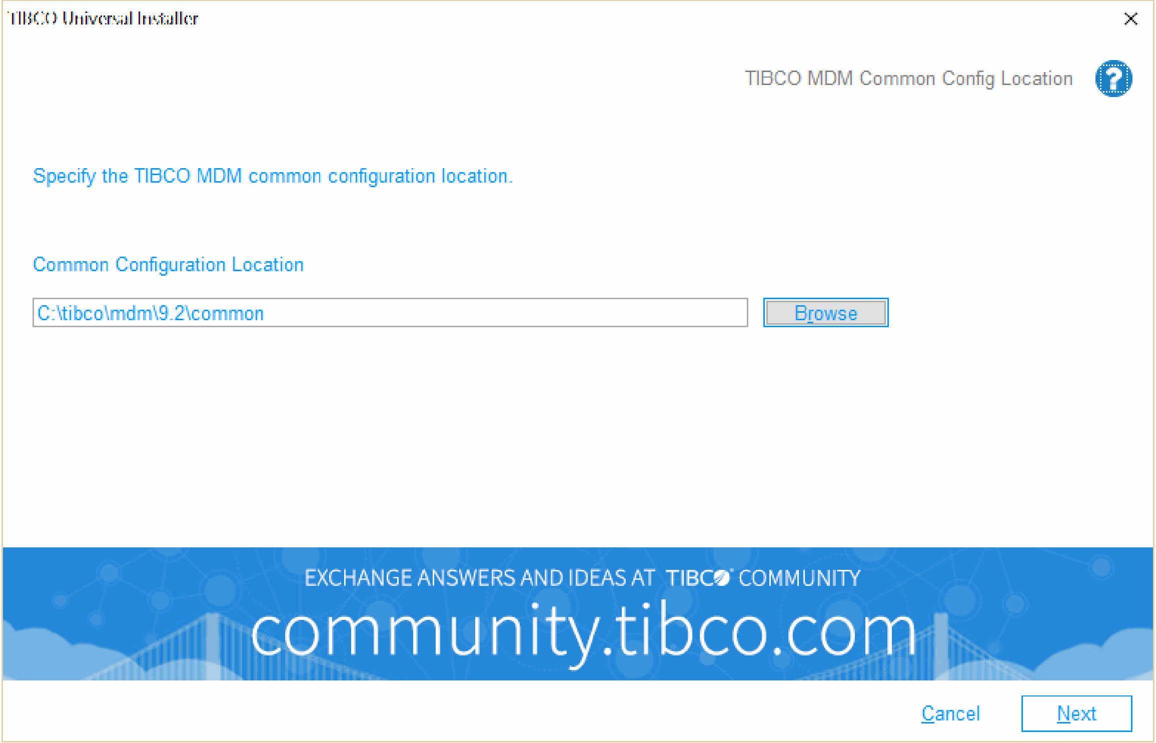
5. Click **Next**. The Installation Profile Selection window is displayed. By default, the **Typical** installation profile is selected.



a) If you select the **Customize Installation** check box, a list of components is enabled (**Executable Image** and **Common Configuration**).

b) Select your preferred options and click **Next**.

6. In the TIBCO MDM Common Config Location window, select the common configuration location and click **Next**.


TIBCO Universal Installer
TIBCO MDM Common Config Location
EXCHANGE ANSWERS AND IDEAS AT TIBCO COMMUNITY
community.tibco.com
Browse
Cancel
Next

Specify the TIBCO MDM common configuration location.

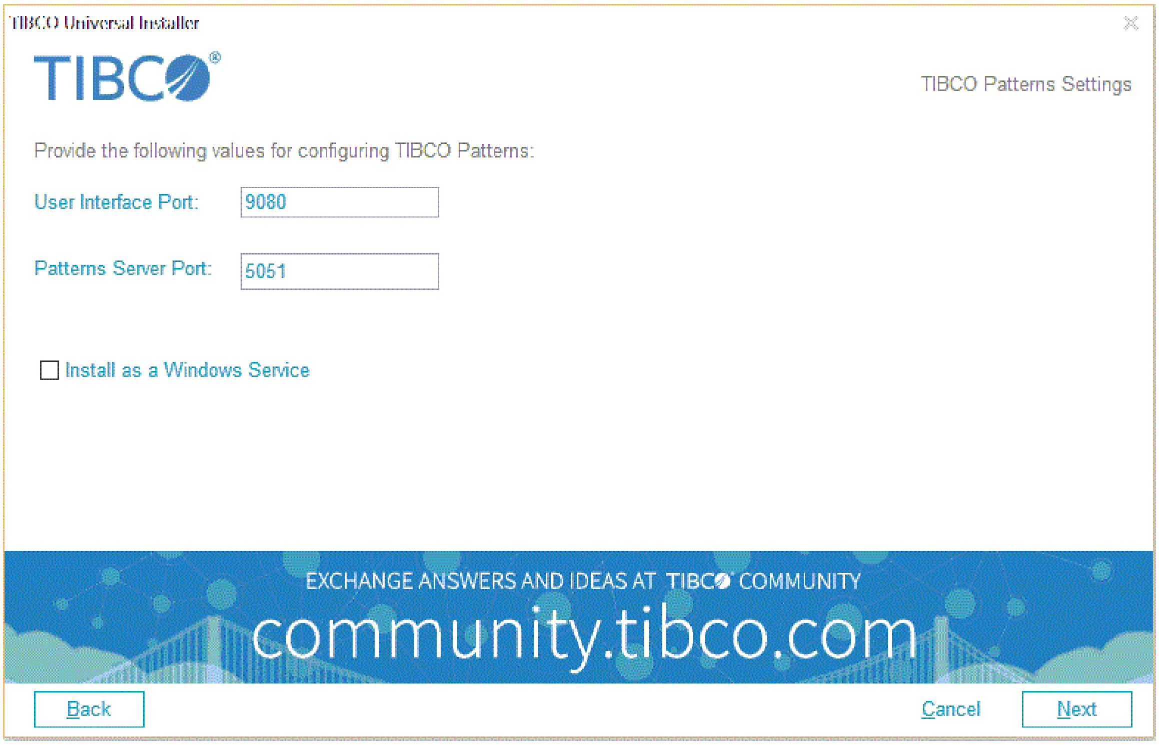
Common Configuration Location

C:\tibco\mdm\9.2\common

If you have selected the standard common configuration option, the default `$MQ_HOME\common` directory location is displayed. You can retain the location or change it by clicking **Browse**.

> If you do not specify the common configuration location, Configurator does not start and you need to perform the manual configuration. For information, refer to Manually Configuring TIBCO MDM.

7. In the TIBCO Patterns Settings window, the default user interface port value (required for Patterns GUI tomcat server) and Patterns server port value (required for Patterns binary) are displayed. You can change the default port values.



TIBCO Universal Installer

TIBCO Patterns Settings

Provide the following values for configuring TIBCO Patterns:

User Interface Port: 9080

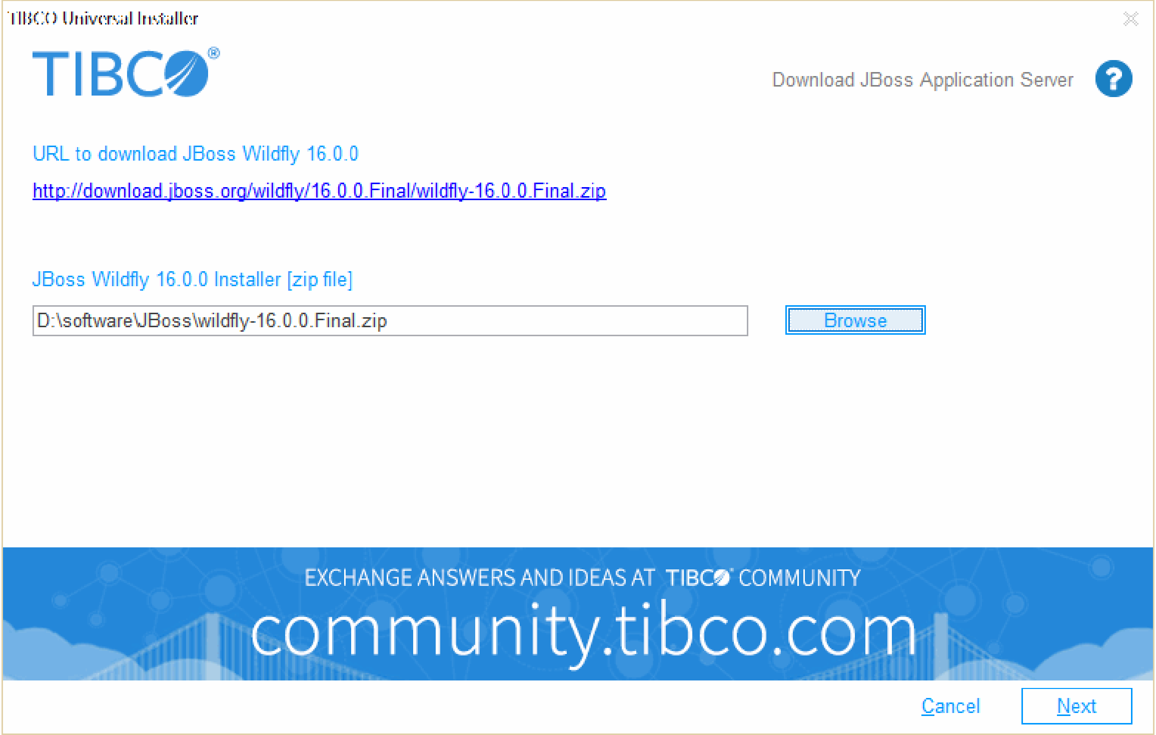
Patterns Server Port: 5051

☐ Install as a Windows Service

a) Select **Install as a Windows Service** check box if you want to install the Patterns server binary as a service.

Click **Next**. The installation process starts.

8. In the Download JBoss Application Server window, do one of the following:



TIBCO Universal Installer

Download JBoss Application Server

URL to download JBoss Wildfly 16.0.0

http://download.jboss.org/wildfly/16.0.0.Final/wildfly-16.0.0.Final.zip

JBoss Wildfly 16.0.0 Installer [zip file]

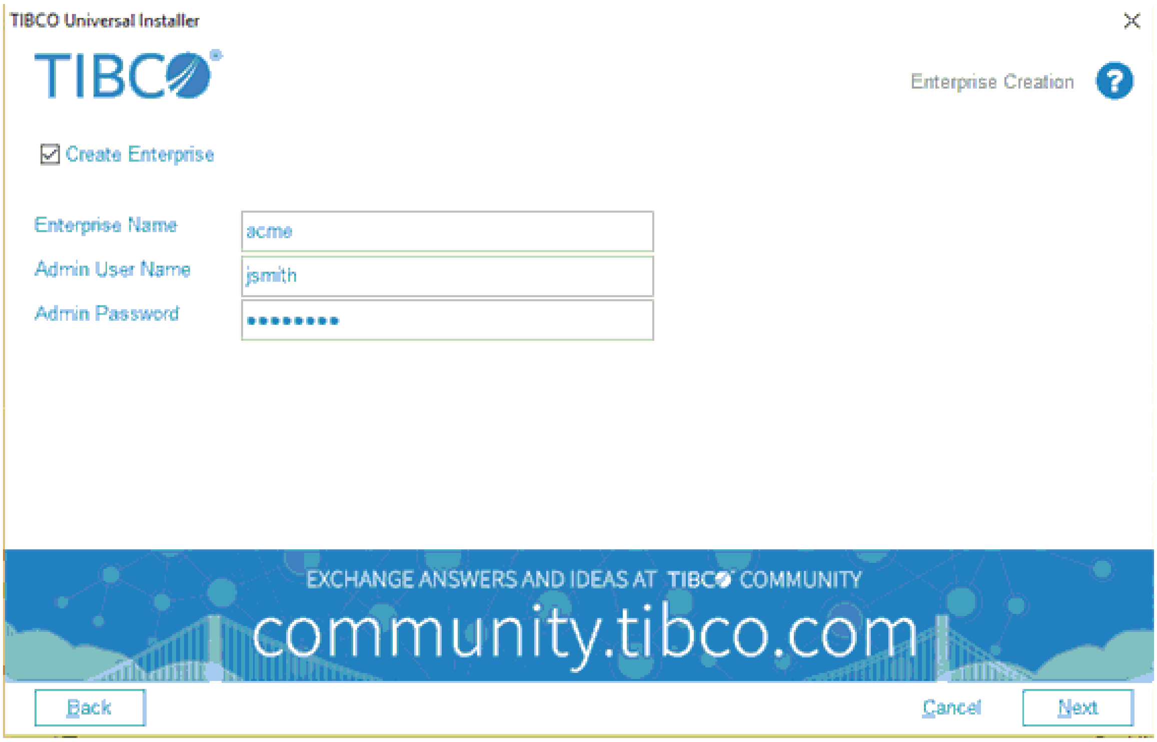
D:\software\JBoss\wildfly-16.0.0.Final.zip

a) Download the `wildfly-16.0.0.Final.zip` file by clicking the URL and save it to your local drive.

b) Click **Browse** and select the ZIP file location. If you have already downloaded the JBoss Application Server, point it to the existing ZIP file and click **Next**.

> The installer validates the `wildfly-appclient-16.0.0.Final.jar` file located in the `wildfly-16.0.0.Final/modules/system/layers/base/org/jboss/as/appclient/main` directory.

9. In the Enterprise Creation window, do the following:


TIBCO Universal Installer
TIBCO
Enterprise Creation
EXCHANGE ANSWERS AND IDEAS AT TIBCO COMMUNITY
community.tibco.com
Back
Cancel
Next

☑ Create Enterprise

Enterprise Name: acme

Admin User Name: jsmith

Admin Password: •••••••••

a) Select the **Create Enterprise** check box. The following fields are enabled:

**Enterprise Name**: Type the enterprise name.

**Admin User Name**: Type the user name of the administrator.

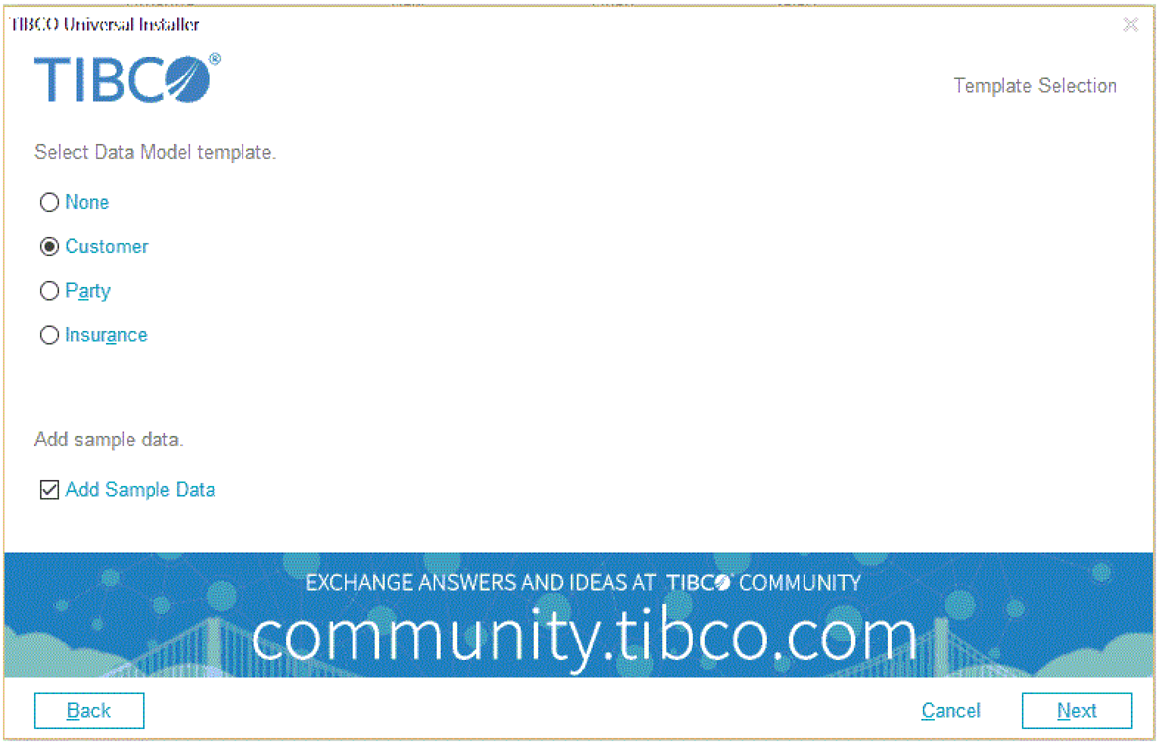
**Admin Password**: Type the password of the administrator.

> The password must contain eight alphanumeric characters with at least one lowercase and one uppercase alphabets and one numeric entry. Additionally, if you want to add special characters, change the default value of the `Password Required` property. For information on password policy, refer to the section, "Configuration Properties for Password" in *TIBCO MDM System Administration*.

Click **Next**.

> If you do not select the **Create Enterprise** check box, an enterprise and administrator credentials fields are disabled, and the next Template Selection window is not displayed.

10. In the Template Selection window, do the following:



TIBCO Universal Installer

Template Selection

Select Data Model template.

○ None

● Customer

○ Party

○ Insurance

Add sample data.

☑ Add Sample Data

a) Select any one of the data models:

- **None**

  If you select **None**, the **Customer** , **Party** and **Insurance** options and the **Add Sample Data** check box are disabled.

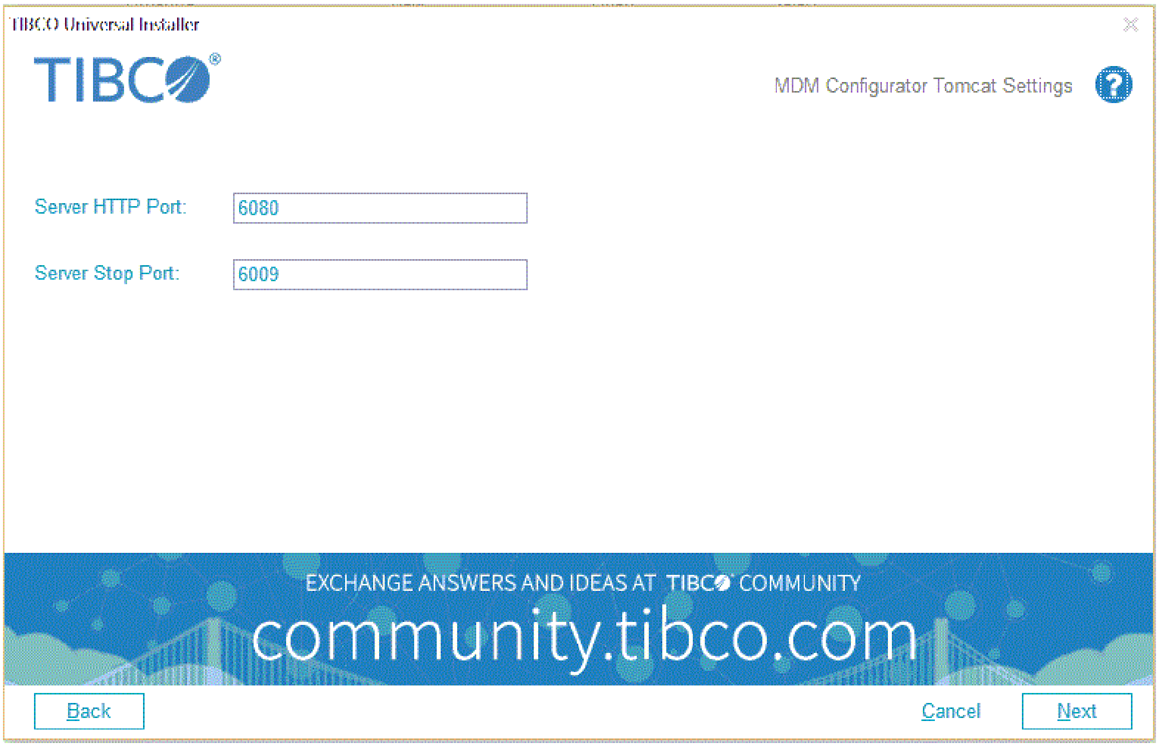
- **Customer**
- **Party**
- **Insurance**

b) If you select **Customer** or **Party** or **Insurance** option, you can also select the **Add Sample Data** check box.

c) However, if you do not select the **Add Sample Data** check box, the customer, party or insurance models are imported but the sample data is not added.

d) Click **Next**.

11. In the MDM Configurator Tomcat Settings window, the default port values are displayed. You can change the default port values.


TIBCO Universal Installer
TIBCO
MDM Configurator Tomcat Settings
EXCHANGE ANSWERS AND IDEAS AT TIBCO COMMUNITY
community.tibco.com
Back
Cancel
Next

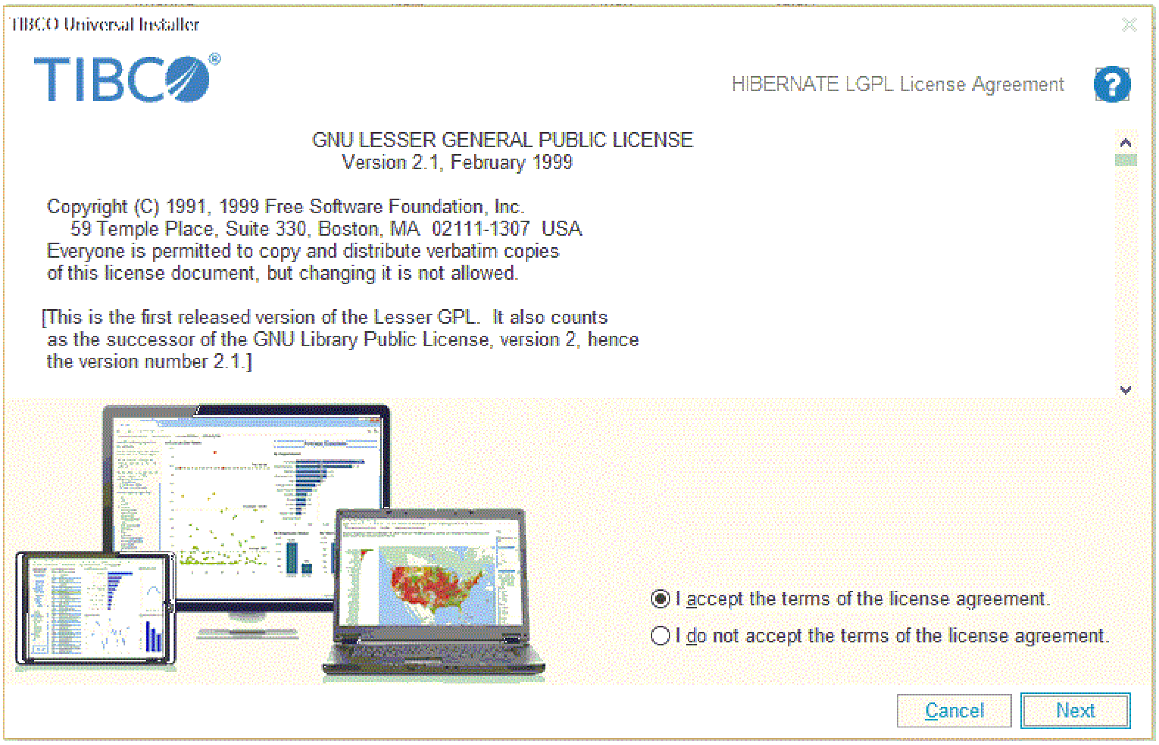
Server HTTP Port: 6080

Server Stop Port: 6009

Click **Next**.

The specified port values must not be in use by other application.

12. The HIBERNATE LGPL License Agreement window is displayed.



GNU LESSER GENERAL PUBLIC LICENSE
Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

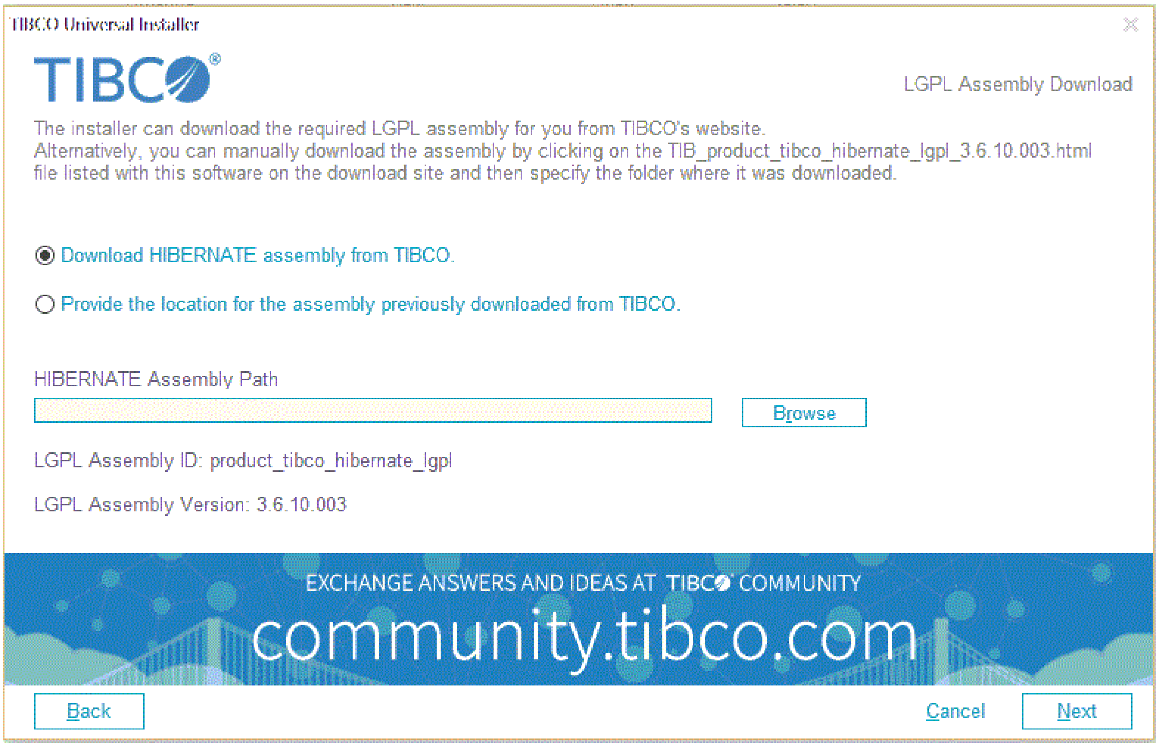
[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

● I accept the terms of the license agreement.

○ I do not accept the terms of the license agreement.

If you agree with its terms, accept the license agreement, and click **Next**.

13. In LGPL Assembly Download window, select one of the following options:



TIBCO Universal Installer

LGPL Assembly Download

The installer can download the required LGPL assembly for you from TIBCO's website.
Alternatively, you can manually download the assembly by clicking on the TIB_product_tibco_hibernate_lgpl_3.6.10.003.html file listed with this software on the download site and then specify the folder where it was downloaded.

● Download HIBERNATE assembly from TIBCO.

○ Provide the location for the assembly previously downloaded from TIBCO.

HIBERNATE Assembly Path

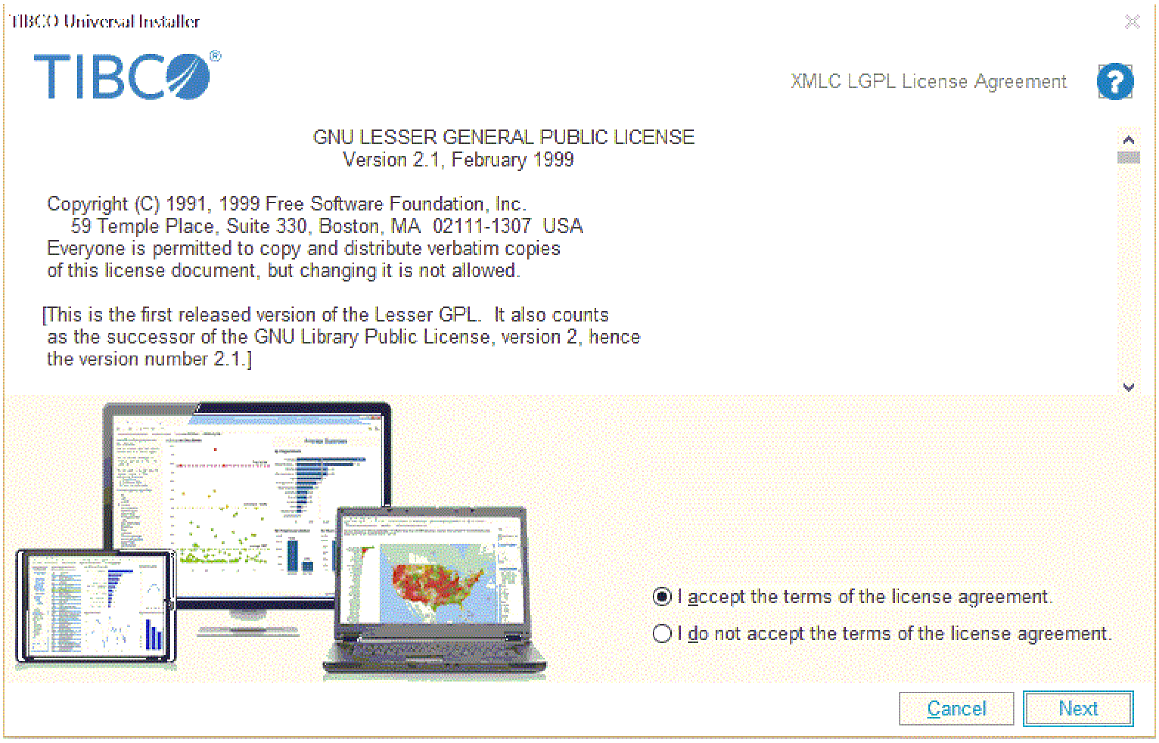
LGPL Assembly ID: product_tibco_hibernate_lgpl

LGPL Assembly Version: 3.6.10.003

- **Download HIBERNATE assembly from TIBCO**: if you select this option, the HIBERNATE assembly is downloaded and configured.
- **Provide the location for the assembly previously downloaded from TIBCO**: If you select this option, the **HIBERNATE Assembly Path** field is enabled. Click **Browse** to select the HIBERNATE assembly.

Click **Next**.

14. The XMLC LGPL License Agreement window is displayed.



TIBCO Universal Installer

XMLC LGPL License Agreement

GNU LESSER GENERAL PUBLIC LICENSE
Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

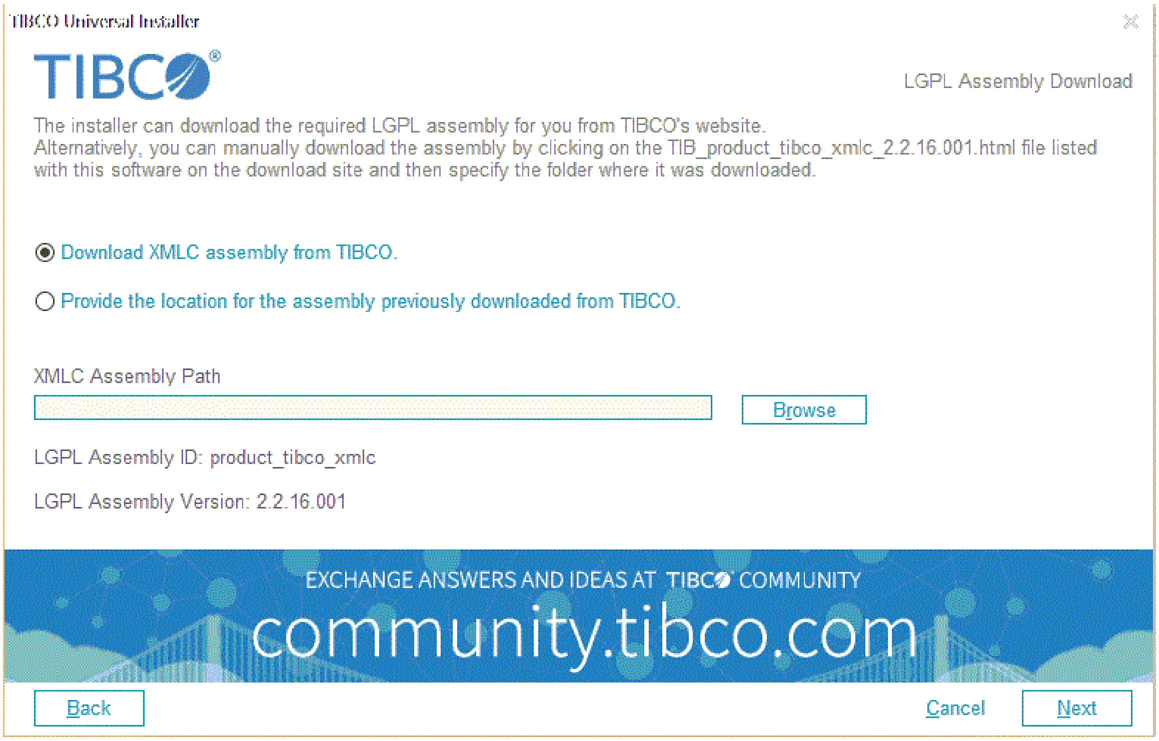
[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

● I accept the terms of the license agreement.

○ I do not accept the terms of the license agreement.

If you agree with its terms, accept the license agreement, and click **Next**.

15. In LGPL Assembly Download window, select one of the following options:



TIBCO Universal Installer

LGPL Assembly Download

The installer can download the required LGPL assembly for you from TIBCO's website.
Alternatively, you can manually download the assembly by clicking on the TIB_product_tibco_xmlc_2.2.16.001.html file listed with this software on the download site and then specify the folder where it was downloaded.

● Download XMLC assembly from TIBCO.

○ Provide the location for the assembly previously downloaded from TIBCO.

XMLC Assembly Path

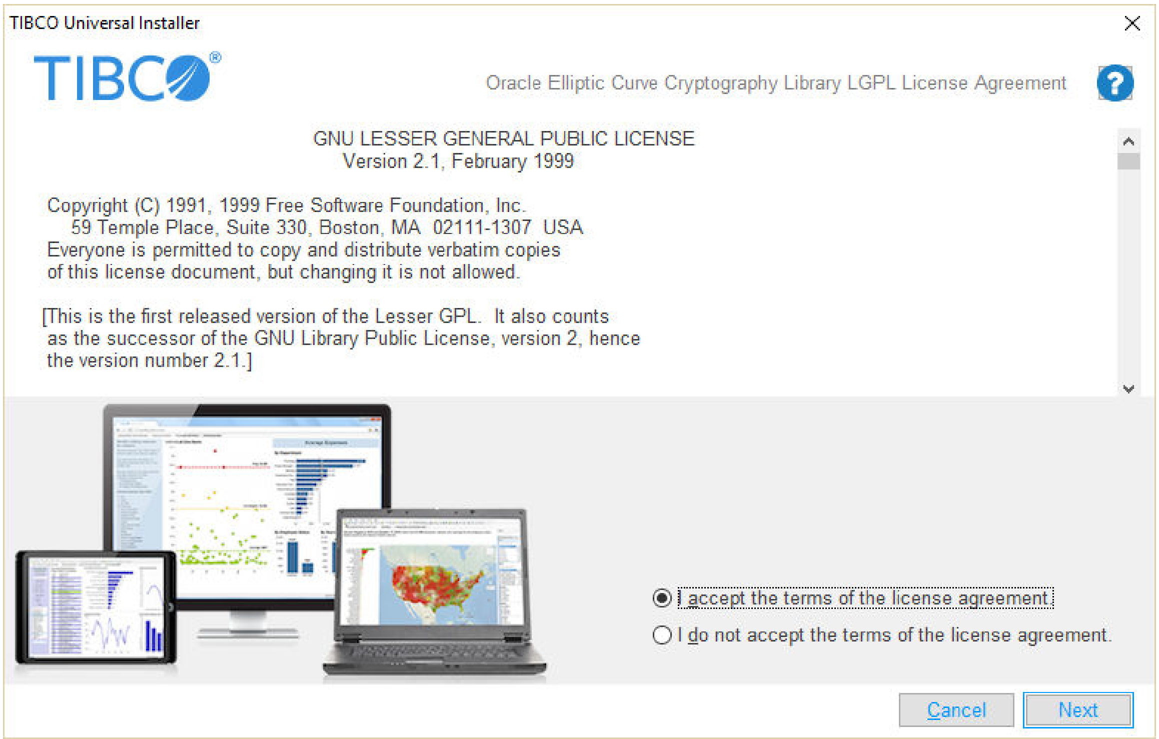
LGPL Assembly ID: product_tibco_xmlc

LGPL Assembly Version: 2.2.16.001

- **Download XMLC assembly from TIBCO**: if you select this option, the XMLC assembly is downloaded and configured.
- **Provide the location for the assembly previously downloaded from TIBCO**: If you select this option, the **XMLC Assembly Path** field is enabled. Click **Browse** to select the XMLC assembly.

Click **Next**.

16. The Oracle Elliptic Curve Library LGPL License Agreement window is displayed.



TIBCO Universal Installer

Oracle Elliptic Curve Cryptography Library LGPL License Agreement

GNU LESSER GENERAL PUBLIC LICENSE
Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

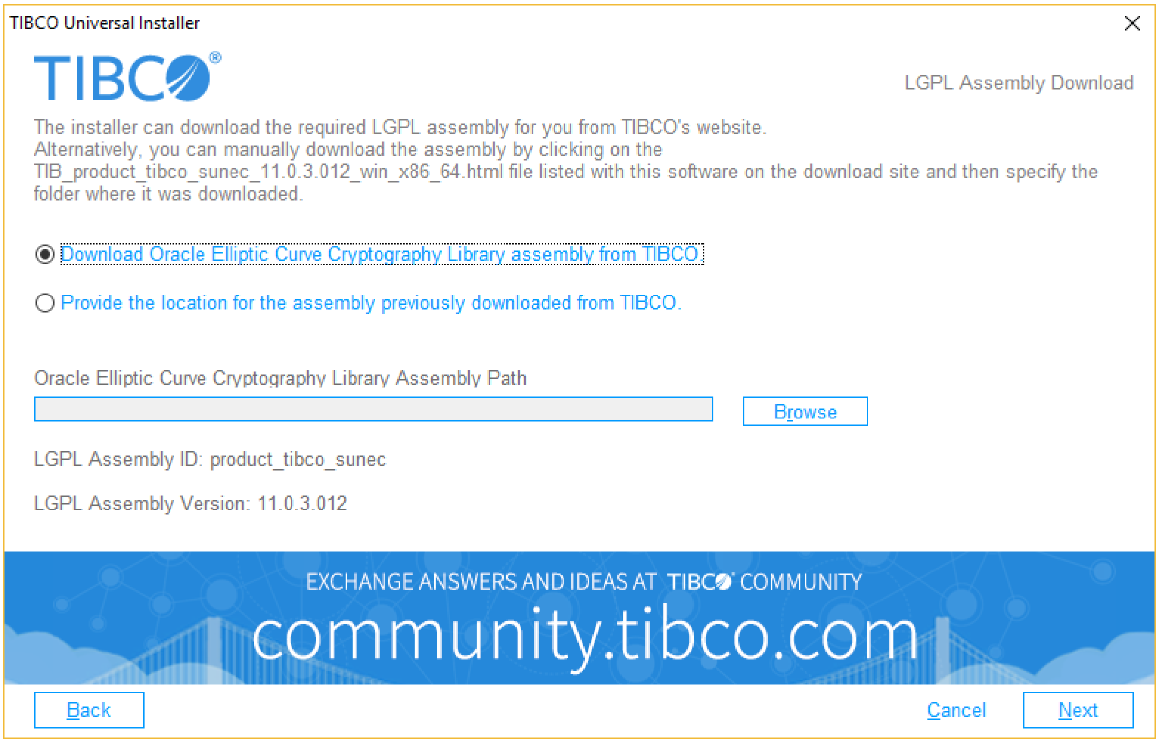
[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

● I accept the terms of the license agreement

○ I do not accept the terms of the license agreement.

If you agree with its terms, accept the license agreement, and click **Next**.

17. In LGPL Assembly Download window, select one of the following options:



TIBCO Universal Installer

LGPL Assembly Download

The installer can download the required LGPL assembly for you from TIBCO's website.
Alternatively, you can manually download the assembly by clicking on the
TIB_product_tibco_sunec_11.0.3.012_win_x86_64.html file listed with this software on the download site and then specify the folder where it was downloaded.

● Download Oracle Elliptic Curve Cryptography Library assembly from TIBCO

○ Provide the location for the assembly previously downloaded from TIBCO.

Oracle Elliptic Curve Cryptography Library Assembly Path

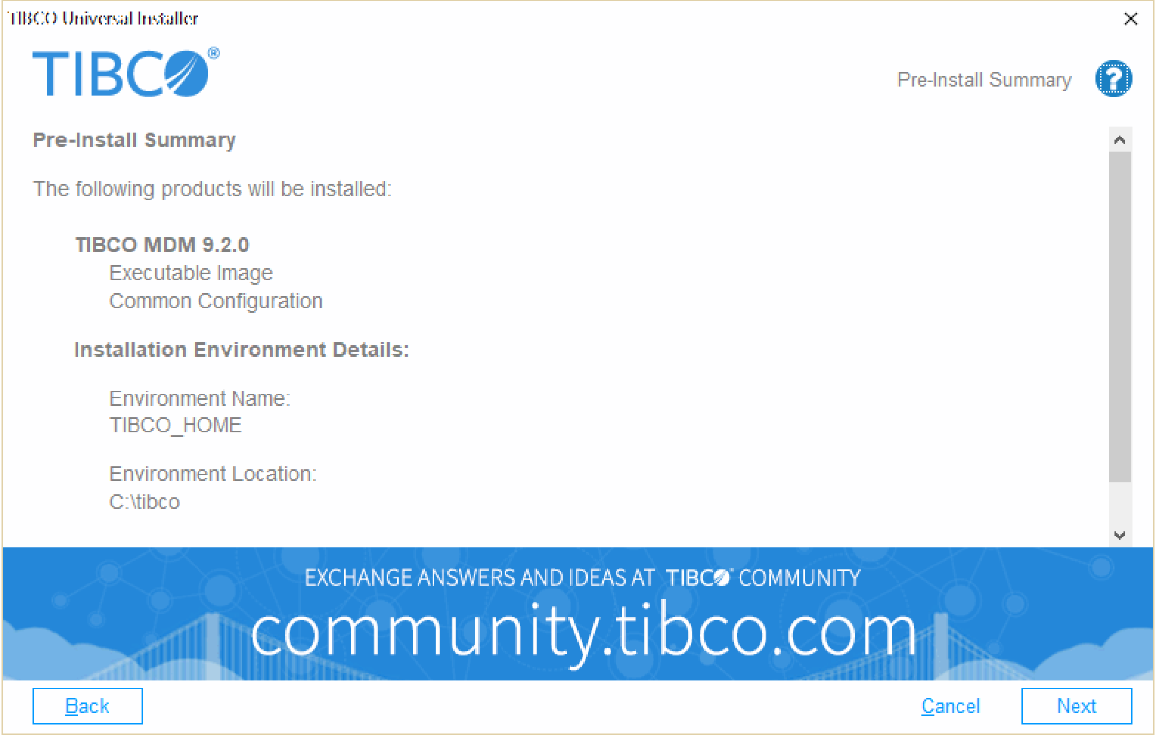
LGPL Assembly ID: product_tibco_sunec

LGPL Assembly Version: 11.0.3.012

- **Download Oracle Elliptic Curve Cryptography Library assembly from TIBCO**: if you select this option, the Cryptography assembly is downloaded and configured.
- **Provide the location for the assembly previously downloaded from TIBCO**: If you select this option, the **Oracle Elliptic Curve Cryptography Library Assembly Path** field is enabled. Click **Browse** to select the Cryptography assembly.

Click **Next**.

18. In the Pre-Install Summary window, review a list of the components that are going to be installed and the installation environment details and click **Next**.


TIBCO Universal Installer
TIBCO
Pre-Install Summary
EXCHANGE ANSWERS AND IDEAS AT TIBCO COMMUNITY
community.tibco.com
Back
Cancel
Next

Pre-Install Summary

The following products will be installed:

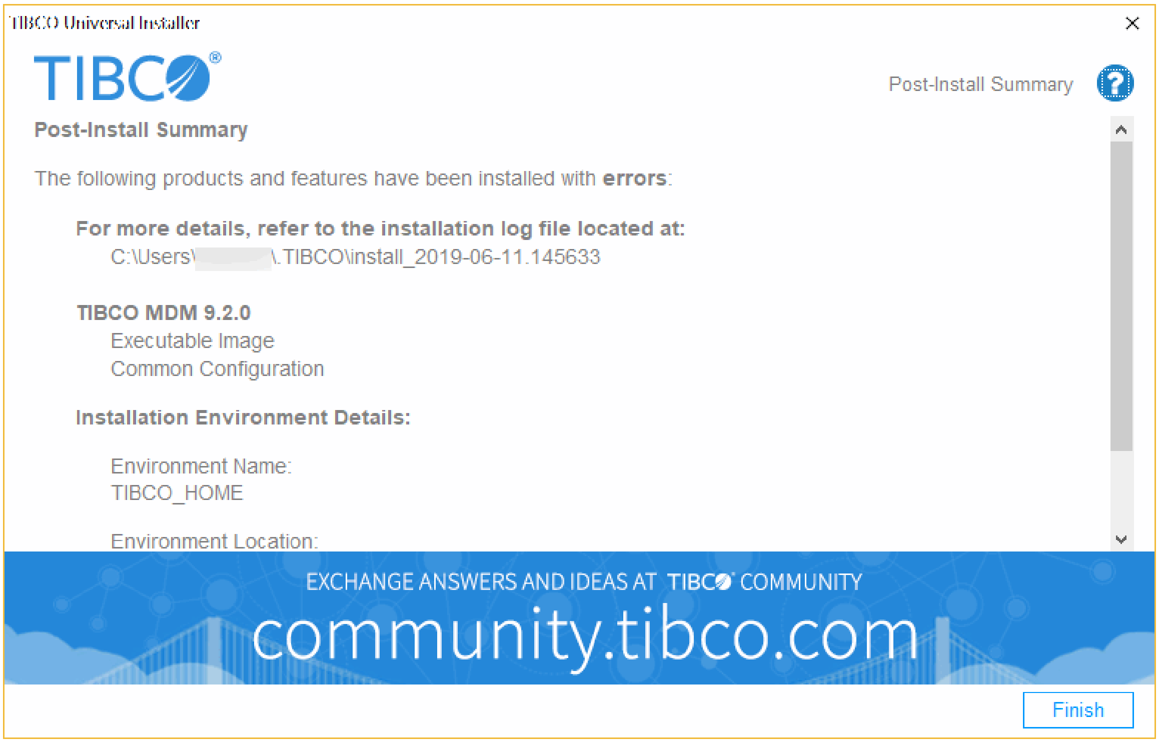
TIBCO MDM 9.2.0
Executable Image
Common Configuration

Installation Environment Details:

Environment Name:
TIBCO_HOME

Environment Location:
C:\tibco

19. In the Post-Install Summary window, review a list of the components that are installed and the installation environment details.



TIBCO Universal Installer
Post-Install Summary
EXCHANGE ANSWERS AND IDEAS AT TIBCO COMMUNITY
community.tibco.com
Finish

**Post-Install Summary**

The following products and features have been installed with **errors**:

**For more details, refer to the installation log file located at:**
C:\Users\[redacted]\.TIBCO\install_2019-06-11.145633

**TIBCO MDM 9.2.0**
Executable Image
Common Configuration

**Installation Environment Details:**

Environment Name:
TIBCO_HOME

Environment Location:

20. Click **Finish** to exit the wizard.

> On some Windows environments, database may not get created postinstallation. For successful database creation, create a folder, apply the Full Control permission to it, and then install TIBCO MDM in this specific folder.

### Performing Postinstallation Tasks for Simple Installation

For the first time installation, the MDM server, netrics server, and Configurator automatically starts. You need to complete the postinstallation tasks for the second time installation of TIBCO MDM. After installation, the `$MQ_HOME` contains most of the required softwares.

| Folder | Component |
|---|---|
| `$MQ_HOME/bin/as` | ActiveSpaces Cache Server |
| `$MQ_HOME/bin/wildfly-`*`version`*`.Final` | JBoss WildFly Application Server |
| `$MQ_HOME/bin/pgsql` | PostgreSQL Database Engine |
| `$TIBCO_HOME/tps` | TIBCO Patterns Server |

The following table lists the ports available after installation:

| Port Number | Component |
|---|---|
| 8080 | HTTP JBoss WildFly Application Start Server Port |
| 8009 | HTTP JBoss WildFly Application Stop Server Port |

| Port Number | Component |
|---|---|
| 6080 | Configurator HTTP Port |
| 5051 | TIBCO Patterns Server Standard Port |
| 5432 | PostgreSQL Database Standard Port |
| 8817 | ActiveSpaces Cache Server Default Port |

### Procedure

1. Start TIBCO MDM Server: Ensure that`$JAVA_HOME` is set.
   a) Select the **Start MDM Server** option from the Programs menu or run **`StartMDMServer.bat(.sh)`**. The file is located at `$MQ_HOME/bin/wildfly-`*`version`*`.Final/bin`.

      The TIBCO MDM server starts. Starting the TIBCO MDM server starts the PostgreSQL database instance and netricsServer.
   b) Optional: If you want to stop the TIBCO MDM server, select the **Stop MDM Server** option from the Programs menu or run **`StopMDMServer.bat(.sh)`**. The file is located at `$MQ_HOME/bin/wildfly-`*`version`*`.Final/bin`.

      The TIBCO MDM server stops. Stopping the TIBCO MDM server stops the PostgreSQL database instance and netricsServer.
2. Start TIBCO MDM: Type http://localhost:8080/eml/Login in the browser. For more information on accessing TIBCO MDM, see "Introduction to TIBCO MDM" chapter of the *TIBCO MDM User's Guide*.
3. Optional: Start and Stop PostgreSQL Server
   a) To start PostgreSQL, run **`pg_start.bat`** or **`pg_start.sh`**.
   b) To stop PostgreSQL, run **`pg_stop.bat`** or **`pg_stop.sh`**.

      The files are located at `$MQ_HOME/db/postgreSQL/install`.
4. Optional: Start and Stop TIBCO Patterns Server
   a) To start the Patterns Server, run **`netricsServer.bat -startService`** or **`netricsServer.sh -startService`**.
   b) To stop the Patterns Server, run **`netricsServer.bat -stopServer`** or **`netricsServer.sh -stopServer`**.

      The files are located at `$MQ_HOME/bin`.

      For more information, see *TIBCO MDM System Administration*.
5. Optional: Start and Stop Configurator

   If you want to change default configuration, you can start the configurator. Ensure that the `$JAVA_HOME` environment variable is set and it points to the valid JDK*version* installation path.
   a) Use the **Start Server** option or run startup.bat/.sh to start the Configurator.
   b) Use the **Shutdown Server** option or run shutdown.bat/.sh to stop the Configurator.

      The `startup.bat/.sh` and `shutdown.bat/.sh` files are located at `$MQ_HOME/configurator/tomcat/bin`.
   c) Use the **Launch** option or double-click `$MQ_HOME/configurator/launch.html` to start Configurator.
   d) To log into Configurator, type `admin` as user name and password credentials.

## Troubleshooting with Simple Installation

If you encounter an issue while installing TIBCO MDM with PostgreSQL, you may resolve the issues by completing the common troubleshooting procedures.

*Troubleshooting with Simple Installation*

| Issue | Description | Solution |
|---|---|---|
| Database specific error | The following exception is displayed in `$MQ_HOME/log/elink.log`: `Unable to get managed connection for java:jboss/eCMDataSource`<br>The exception may get display due to the following reasons:<br>• PostgreSQL is not running.<br>• PostgreSQL initialization failed.<br>• MDM database schema does not exist. | To resolve this error, perform the following steps:<br>Execute **`pg_start.bat/.sh`** and **`pg_stop.bat/.sh`** to start and stop PostgreSQL Server. The files are located at `$MQ_HOME/db/postgreSQL/install`.<br>Execute **`pg_init.bat/.sh`** to initialize the PostgreSQL database instance. The files are located at `$MQ_HOME/bin/pgsql`.<br>Perform the following tasks:<br>• Create the missing tablespace directories:<br>`$MQ_HOME/bin/pgsql/tablespaces/velodbdata $MQ_HOME/bin/pgsql/tablespaces/velodbindx`<br>Ensure that the logged in user is the owner and has full permission to access these directories.<br>• Execute **`install.bat`** from `$MQ_HOME/db/postgreSQL/install` and type the following:<br>`$MQ_HOME/bin/pgsql localhost 5432 postgres` *`USERNAME`* `"" mdmuser mdmpassword`<br>Where *USERNAME* is the logged-in user name.<br>`$MQ_HOME/bin/pgsql/tablespaces/velodbdata $MQ_HOME/bin/pgsql/tablespaces/velodbindx dev dev`<br>Verify if the MDM schema exists by starting the PostgreSQL Administration console. |

<table>
<tr><th>Issue</th><th>Description</th><th>Solution</th></tr>
<tr>
<td>Other ActiveSpaces servers with different versions communicating with each other</td>
<td>The following warning message is displayed in the JBoss Console: <code>Message received from older version or unknown source, bad magic_number</code></td>
<td>Modify <code>com.tibco.cim.cache.as.discoveryurl</code> and <code>com.tibco.cim.cache.as.listenurl</code> properties specified in the <code>ConfigValues.xml</code> file as follow:
<pre>
<ConfValue description="Provide
the unicast discovery urls for
AS." name="AS unicast discovery
URLS"
propname="com.tibco.cim.cache.as
.discoveryurl"
sinceVersion="8.2"
visibility="Advanced">
  <ConfList>
    <ConfListString
value="127.0.0.1:8817"/>
  </ConfList>
</ConfValue>
<ConfValue description="Provide
the listen url for AS."
name="AS Listen URL"
propname="com.tibco.cim.cache.as
.listenurl" sinceVersion="8.2"
visibility="Advanced">
  <ConfString default="tcp://"
value="tcp://127.0.0.1:8817"/></
ConfValue>
</pre>
</td>
</tr>
</table>

## Installing TIBCO MDM (Typical Installation)

In the typical installation, the installer presents panels which you can select choices about the product location, and so on.

**Prerequisites**

- Verify that your computer meets the System requirements. The system requirements are listed in the `readme.txt` file.
- Download the Installer. The application is available as a ZIP file.
- Extract the contents of the ZIP file to a folder on your computer.
- Search for the TIBCOUniversalInstaller application in the location where you have extracted the ZIP file.

### Procedure

1. Run the `TIBCOUniversalInstaller.exe` application. The TIBCO Universal Installer Welcome window is displayed.
2. Review the information and click **Next**. The license agreement is displayed.
3. Review the terms of the license agreement. If you agree with its terms, accept the license agreement and click **Next**.
4. In the Installation Profile Selection window, do one of the following:
   a) Select **Create a new TIBCO_HOME** option. A TIBCO installation environment is used for software installations and consists of a Name and Directory. Products installed into different installation environments do not share components; therefore you can keep product installations completely isolated from each other.

- **Directory**: browse to the directory where product needs to be installed and provide a unique environment name. Ensure that you have write permission to this directory.
- **Name**: specify the environment name that is easy to identify your environment. For example, 'User Acceptance' or 'Procurement Department'.

a) If you have previously installed a TIBCO product using the Universal Installer, you can select **Use an existing TIBCO_HOME**. By default, the installer detects the directory for your TIBCO_HOME and displays the path. For example, on Windows, the default installation directory is `c:\tibco`.

> If you had installed the application earlier, the directory cannot be modified. For a new installation, the directory can be changed.

Click **Next**.

5. In the Installation Profile Selection window, by default, the **Typical** installation profile is selected.

   a) If you check the **Customize Installation** check box, a list of components is enabled (**Executable Image** and **Common Configuration**).

   b) Select your preferred options and click **Next**.

6. In the MDM Configurator Tomcat Settings window, the default port values are displayed. You can change the default port values. Click **Next**.

   > The specified port values must not be in use by another application. If you do not specify configuration information, Configurator does not start and you need to perform manual configuration. For more information on Configurator, refer to Configurator.

7. In the TIBCO Patterns Settings window, the default user interface port value (required for Patterns GUI tomcat server) and Patterns server port value (required for Patterns binary) are displayed. You can change the default port values.

   a) Select **Install as a Windows Service** check box if you want to install the Patterns server binary as a service.

   Click **Next**.

8. In the TIBCO MDM Common Config Location window, select the common configuration location and click **Next**.

   If you have selected the standard common configuration option, the default `$MQ_HOME\common` directory location is displayed. You can retain the location or change it by clicking **Browse**.

   > The best practice is to assign a location separate from `MQ_HOME`. TIBCO MDM creates files in this directory and it is better to keep it separate from `MQ_HOME`. In a clustered environment, this directory should be shared for all instances.

   > If you do not specify the common configuration location, Configurator does not start and you need to perform the manual configuration. For information, refer to Manually Configuring TIBCO MDM.

9. The HIBERNATE LGPL License Agreement window is displayed. If you agree with its terms, accept the license agreement, and click **Next**.

10. The LGPL Assembly Download window is displayed. Select one of the following options:

    - **Download HIBERNATE assembly from TIBCO**: if you are downloading the LGPL assembly for the first time, select this option. The HIBERNATE assembly is downloaded in the same folder as the installer. All the third party software which are used in the application and their licenses are downloaded.
    - **Provide the location for the assembly previously downloaded from TIBCO**: if you have previously downloaded the LGPL assembly, specify the folder in which you have downloaded the hibernate assembly. Browse to the directory where the assembly is previously downloaded and saved.

> If the LGPL Assembly is already downloaded, the LGPL Assembly Download window is not displayed.

Click **Next**.

11. The XMLC LGPL License Agreement window is displayed. If you agree with its terms, accept the license agreement, and click **Next**.
12. Select the **Download XMLC assembly from TIBCO** option. The assembly gets downloaded in the same folder as the installer. Click **Next**.
13. In the Pre-Install Summary window, review a list of the components that are going to be installed and the installation environment details and click **Next** to begin the installation process.
14. In the Post-Install Summary window, review a list of the components that are installed and the installation environment details. Click **Finish** to exit the wizard.

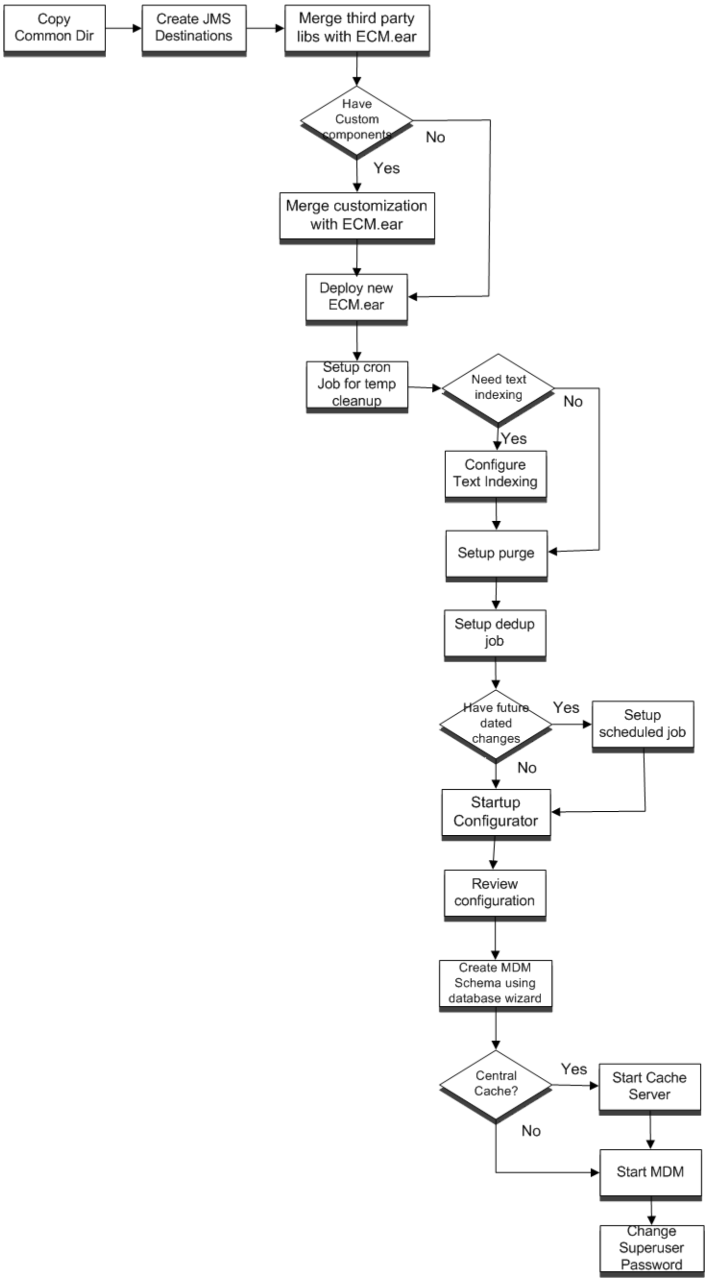
### Result

> The following auto generated log files are created in the `C:\Users\`*username*.TIBCO\install_*currentyear-currentmonth-currentdate.uniqueID* folder: Using the log files, you can troubleshoot for errors in the installation process.
>
> - antTask_log_installerConfig_*currentyear-currentmonth-currentdate.uniqueID*: consists configuration related logs.
> - antTask_log_installerMergeXMLC_*currentyear-currentmonth-currentdate.uniqueID*: consists the XMLC merge related logs.
> - tibco_universal_installer.*username*_install: consists installer related logs.
> - antTask_log_updateEARToIncludeHibernateLib_*currentyear-currentmonth-currentdate.uniqueID*: consists `ECM.ear` file related logs.
> - antTask_log_copyASFolderToBin_*currentyear-currentmonth-currentdate.uniqueID*:: consists ActiveSpaces folder related logs.

## Postinstallation Task Roadmap for Typical Installation

After you install TIBCO MDM, you must complete other tasks, such as merging third party libraries with ECM.ear, deploying the updated ECM.ear, and so on.

Refer to the following diagram for step by step post installation tasks:



## Performing Postinstallation Tasks for Typical Installation

You need to merge third party libraries with `ECM.ear` that you have created for the WebSphere and WebLogic application servers and manually copy TIBCO EMS libraries that are not shipped with the product due to Licensing restrictions.

### Procedure

1. Merge Third Party Libraries with `ECM.ear`.
   a) Go to `$MQ_HOME/build/custom`.
   b) Execute `customUtil.bat` or `customUtil.sh` **`-mergeExternalLibrary`**.

      This command creates the thirdPartyLibrary folder in `$MQ_HOME`.
   c) Copy the consolidated third party JAR file (`ThirdParty.jar`) to this folder and enter **y** to proceed for merging. For information on creating a consolidated JAR file, refer to Creating a Consolidated JAR File with Third Party Libraries.

Continue and complete the script. The updated `ECM.ear` is placed in `$MQ_HOME`.

2. Copy EMS Libraries.

   a) Copy the `tibjms.jar` and `jms-2.0.jar` from `$EMS_HOME/lib` to `$MQ_HOME/lib/external` directory.

⚠ If you do not copy the `tibjms.jar` and `jms-2.0.jar` files in `$MQ_HOME/lib/external` directory, the utilities do not work.

## Troubleshooting with Typical Installation

If you encounter an issue while installing TIBCO MDM, you may resolve the issues by completing the common troubleshooting procedures.

***Troubleshooting with Typical Installation***

| Issue | Description | Solution |
|---|---|---|
| Display Cannot be Opened | The UNIX installer terminates with the following error message:<br>"Could not open display" | If you run the UNIX installer in a graphical mode, your local display has to be specified to the server. This local display needs to have a X Windows client installed. The local display is typically indicated to the server by specifying the DISPLAY environment variable. For example, on a bash shell:<br>`export DISPLAY=`*`client host name`*`:0.0`<br>The X windows client also requires that the connection from the server is authorized. To enable the authorization, refer to the X Windows client documentation. On many X Windows clients, the authorization can be granted by using the Xhost command. For example:<br>Xhost + |

| Issue | Description | Solution |
|---|---|---|
| Installer Terminates | The installer terminates without any error message. | The installer creates a detailed timestamped log file in one of the following places:<br>• If `TIBCO_HOME` did not exist at install time, the log file is created in the temp directory of the user in a.TIBCO sub folder.<br>• If `TIBCO_HOME` existed, the log file is created in the `$TIBCO_HOME/log` folder.<br>Check the log file for any errors and then contact TIBCO Technical Support.<br>You can also run the installer with the installer log enabled using the –is:log option. For example:<br>./TIBCOUniversalInstaller-lnx-x86.bin –is:javahome –is:log *path*/log.dat |

## Installing in Console Mode

Using the Console mode, you can install the software in a non-Windows environment. The installer prompts you for values.
You can install TIBCO MDM in console mode with the following options:

- Installing with Typical Option for Simple Installation
- Installing with Typical and Custom Options

You can move through the installation process as follows:

**Press key** or **1:** Moves forward in the installer

**Press 2:** Goes back to the previous screen

**Press 3:** Cancels the wizard and exits the installation or uninstallation

### Procedure

1. On the command line, type the following command to launch the TIBCO Installer:

   ```
   $ ./TIBCOUniversalInstaller-lnx-x86-64.bin -console
   ```

   The TIBCO Universal Installer Welcome text is displayed along with the information about products to be installed.
2. Press **1** to continue or **3** to cancel. If you press **1**, the License Agreement text is displayed.
3. Press **1** to continue reading or **2** to skip reading. If you press **1**, read through the entire license agreement using the **Enter** key. If you press **2**, the following question is displayed: Do you agree to the terms of the license agreement?
4. Type **Yes** to agree to the license terms or **No** to exit this installer.
5. Press **1** to continue, **2** to go back to the previous panel, or **3** to cancel. If you press **1**, the TIBCO Home Selections options are displayed.

- Press **1** if you want to create a new TIBCO installation environment.
- Press **2** to select an existing environment.

6. If you press **1**, the following options are displayed:

    - Environment Name: Type the environment name. For example, **MDM*version***
    - Environment Location: Type the environment location. For example, **/home/apps/MDM*version***

7. If you press **2**, the existing environment name and locations are displayed:

    - Environment Name: For example, **MDM*version***.
    - Environment Location: For example, **/home/apps/MDM*version***

8. Press **Enter**. The following confirmation message is displayed:

    `Is this the TIBCO Home environment you want to use?`

9. Type **Yes** to confirm.
10. Press **1** to continue, **2** to go back to the previous panel, or **3** to cancel. The Install Profile Selection options are displayed.
11. Press **1** for Typical or **2** for Custom installation.

## Installing with Typical and Custom Options

The Console mode provides two options: Typical and Custom. You can choose to select one of the options to install TIBCO MDM.

### Procedure

1. If you have pressed **1** for Typical install, the Product Feature Selection options along with the selection question are displayed:

    ```
    Product Feature Selections
    --------------------------
       TIBCO MDM version
          [X] Executable Image
          [X] Common Configuration
    Do you wish to proceed with the above feature selections?
    ```

    a) Type **Yes** to proceed or **No** to make changes.
    b) Press **1** to continue, **2** to go back to the previous panel, or **3** cancel.

2. If you have pressed **2** for Custom install, only the Product Feature Selection options are displayed:

    ```
    Choose the features to install.
    1. [ ] Executable Image
    2. [ ] Common Configuration
    ```

    a) Type **1** to select the Executable Image option.
    b) Type **2** to select the Common Configuration option.
    c) Type **0** when you have completed.

3. If you have pressed **1** in the Typical install and **2** in the Custom install, the default TIBCO MDM Common Configuration location is displayed. If you want to change the location, you can change the default path of Common Configuration Location, and then press **Enter**.

    This step is not displayed if you select a custom configuration option and did not opt to install the Common Configuration.

4. Press **1** to continue or **3** to cancel. The LGPL License Agreement text for hibernate assembly is displayed.

5. Press **1** to continue reading or **2** to skip reading. If you press **1**, read through the entire license agreement using the **Enter** key. If you press **2**, the following question is displayed:
   Do you agree to the terms of the license agreement?
6. Type **Yes** to agree to the license terms or **No** to exit this installer. After you agree to the license terms, a screen is displayed with the option whether or not to download hibernate assembly or predownload Hibernate Assembly folder. Based on the selection, the next screen is displayed.
7. Press **1** to continue or **3** to cancel. If you have pressed **1**, the hibernate assemblies are downloaded from https://edelivery.tibco.com. Alternatively, you can opt to select the location for the download.
8. Press **Enter** after the hibernate assembly download is complete. The LGPL License displayed for XMLC library is displayed.
9. Repeat the steps from 5 to 8 to download the XMLC library.

   The Preinstall summary (details about components to be installed along with total size) is displayed.
10. Press **1** to continue, **2** to go back to the previous panel, or **3** to cancel.
    The TIBCO Patterns Settings with the following details are displayed:

    ```
    TIBCO Patterns Settings
    -----------------------
    User Interface Port: [9080]

    Server Port: [5051]

    Install as a Windows Service? [false]
    ```

    The default user interface port value is required for Patterns GUI tomcat server and the Patterns server port value is required for Patterns binary. You can change the default port values. In Linux install, the `Install as a Windows Service` option is displayed with the `false` value.
11. Press **1** to continue or **3** to cancel. If you press **1**, the installation starts and the MDM Tomcat Settings details are displayed.

    While installation is in progress the, TIBCO Universal Installer - MDM Tomcat Settings screen is displayed.
12. Press **Enter** to accept the default Server HTTP Port and Server Stop Port values.

    You can change the default values of the Server HTTP Port and Server Stop Port.
13. Press **1** to continue or **3** to cancel. The postinstallation summary is displayed. Read the postinstallation details.
14. Press **Enter** to exit the installer.

### Installing with Typical Option for Simple Installation

The Typical option of the Console mode provides steps to install TIBCO MDM for simple installation.

#### Procedure

1. If you have pressed **1** for Typical install, the Product Feature Selection options along with the selection question are displayed:

   ```
   Product Feature Selections
   --------------------------
      TIBCO MDM version
         [X] Executable Image
         [X] Common Configuration
   Do you wish to proceed with the above feature selections?
   ```

   a) Type **Yes** to proceed or **No** to make changes.

b) Press **1** to continue, **2** to go back to the previous panel, or **3** cancel.

2. Enter the path of the `WildFly-`*`version`*`.Final` ZIP file.
3. Press **1** to continue or **3** to cancel.
4. If you have pressed **1** in the Typical install and **2** in the Custom install, the default TIBCO MDM Common Configuration location is displayed. If you want to change the location, you can change the default path of Common Configuration Location, and then press **Enter**.

   > This step is not displayed if you select a custom configuration option and did not choose to install the Common Configuration.

5. Press **1** to continue or **3** to cancel.
6. If you have pressed **1**, the enterprise and administrator credentials creation options are displayed:

   ```
   Would you like to create enterprise [Yes]

   If Yes :-
   Enterprise Name:-
   Admin User Name:-
   Admin Password:-
   ```

   a) Type the enterprise name.
   b) Type the user name of the administrator.
   c) Type the password of the administrator.

   > The password must contain eight alphanumeric characters with at least one lowercase and one uppercase alphabets and one numeric entry. Additionally, if you want to add special characters, change the default value of the `Password Required` property. For information on password policy, refer to the section, "Configuration Properties for Password" in *TIBCO MDM System Administration*.

7. Press **1** to continue or **3** to cancel.
8. If you have pressed **1**, the data model selection options are displayed:

   ```
   Choose a data model.
   1. None
   2. Customer
   3. Insurance
   4. Party
   ```

   a) Type **1** to select the `None` option.
   b) Type **2** to select the `Customer` option.
   c) Type **3** to select the `Insurance` option.
   d) Type **4** to select the `Party` option.

9. If you have pressed **2** or **3**, the following message is displayed:

   ```
   Would you like to add sample data ? [Yes]
   ```

   a) Type **Yes** to add the sample data.
   b) Press **1** to continue or **3** cancel.

10. If you have pressed **2** or **3**, the following message is displayed:

    ```
    Provide Configurator Tomcat settings
    Specify Server HTTP Port : [6080]
    Specify Server Stop Port: [6009]
    ```

11. Specify the port values for the Configurator tomcat settings.
12. Press **1** to continue or **3** to cancel.
13. The Common Configuration location is displayed. If you want to change the location, you can change the default path of Common Configuration Location, and then press **Enter**.
14. Press **1** to continue or **3** to cancel.
    The LGPL License Agreement text for hibernate assembly is displayed.

15. Press **1** to continue reading or **2** to skip reading.
    If you press **1**, read through the entire license agreement using the **Enter** key. If you press **2**, the following question is displayed:

```
Do you agree to the terms of the license agreement?
Enter Yes to agree to the license terms or No to exit this installer: Yes
```

16. Type **Yes**.
17. Press **1** to continue or **3** to cancel.
    The LGPL Assembly Download message is displayed:

```
1.Download HIBERNATE assembly from the public TIBCO download site.
2.Specify the location of a previously downloaded HIBERNATE assembly.
Select an option by entering its number from the list above. [1]
```

Depending on the option that you select, LGPL assembly is downloaded.

18. Press **1** to continue, **2** to go back to the previous panel, or **3** to cancel.
    The XMLC LGPL license agreement text is displayed.
19. Press **1** to continue reading or **2** to skip reading.
    If you press **1**, read through the entire license agreement using the **Enter** key. If you press **2**, the following question is displayed:

```
Do you agree to the terms of the license agreement?
Enter Yes to agree to the license terms or No to exit this installer: Yes
```

20. Type **Yes**.
21. Press **1** to continue or **3** to cancel.
    The LGPL XMLC Assembly Download message is displayed.

```
1. Download XMLC assembly from the public TIBCO download site.
2.Specify the location of a previously downloaded XMLC assembly.
Select an option by entering its number from the list above. [1]
```

Depending on the option that you select, LGPL XMLC assembly is downloaded.

22. Type **Yes**.
23. Press **1** to continue, **2** to go back to the previous panel, or **3** to cancel.
    The TIBCO Patterns Settings with the following details are displayed:

```
TIBCO Patterns Settings
-----------------------
User Interface Port: [9080]

Server Port: [5051]

Install as a Windows Service? [false]
```

The default user interface port value is required for Patterns GUI tomcat server and the Patterns server port value is required for Patterns binary. You can change the default port values. In Linux install, the `Install as a Windows Service` option is displayed with the `false` value.

24. Press **1** to continue or **3** to cancel.
    The installation starts and the postinstallation summary is displayed. Read the postinstallation details.
25. Press **Enter** to exit the installer.

## Silent Installation

The TIBCO MDM installer supports silent installation to facilitate automatic installation on to other computers. The silent installation option is based on a silent file which contains all of the information the installer needs to perform the installation.

The Silent Installer is present in the same location as the TIBCO Universal Installer (the location where you extract the TIBCO MDM distributable zip). Search for the `TIBCOUniversalInstaller.SILENT` file.

Open the `TIBCOUniversalInstaller.SILENT` file in a text editor and change the content of the response file to your needs. The separate entries are commented inside the file.

To execute a silent installation, copy the response file and the installer executable into the same directory and use the **`-silent`** option of the installer executable. For example:

Windows

*MDM_Installer_Location*\**`TIBCOUniversalInstaller.exe -silent`**

UNIX

*MDM_Installer_Location*/**`TIBCOUniversalInstaller.bin -silent`**

Where *MDM_Installer_Location* is the path where the installer is extracted.

## Options in the Silent Installer File

Edit the options in the file as required.

The following options are available in the file:

<table>
<thead>
<tr><th>Description of entries in the SILENT Install File</th></tr>
</thead>
<tbody>
<tr><td><pre>
&lt;!--accept the license agreement--&gt;
&lt;entry key="acceptLicense"&gt;true&lt;/entry&gt;
By default, this is set to true for acceptance of the license agreement.
</pre></td></tr>
<tr><td><pre>
&lt;!--If multiple environments are supported, create a new one or use and existing
one--&gt;
&lt;!--If the product does not support multiple environments, then the values below
are ignored--&gt;
&lt;entry key="createNewEnvironment"&gt;true&lt;/entry&gt;
</pre></td></tr>
<tr><td><pre>
&lt;!--If using an existing environment then the installationRoot AND
environmentName MUST match a pre-existing environment--&gt;
&lt;!--If createing a new environment then the installationRoot AND environmentName
MUST BE UNIQUE and not match a pre-existing environment--&gt;
&lt;entry key="environmentName"&gt;TIBCO_HOME&lt;/entry&gt;
&lt;entry key="environmentDesc"&gt;MDM Installation&lt;/entry&gt;
</pre>
By default, the root installation directory is c:\tibco. You can change this provided <code>TIBCO_HOME</code> has not already been set, in which case, the existing <code>TIBCO_HOME</code> is used. Provide the environment name and description. Environment name must exist if using a pre-existing one, and in case of a new one, it must be unique.</td></tr>
<tr><td><pre>
&lt;!--Product Specific Properties can be set below using the same 'entry key='
format as above--&gt;
&lt;entry key="feature_Executable Image_mdm-jboss"&gt;true&lt;/entry&gt;
&lt;entry key="feature_Common Configuration_mdm-jboss"&gt;true&lt;/entry&gt;
&lt;entry key="feature_Documentation_mdm-jboss"&gt;true&lt;/entry&gt;
</pre>
By default, all three components (Executable Image, Common Configuration, and Documentation) are set to <code>true</code> and will be installed. Replace with <code>false</code> for any components that you do not want to install.</td></tr>
<tr><td><pre>
&lt;!-- Specify the absolute path of jboss server zip file--&gt;
&lt;entry key="mdm.jboss.zipfile"&gt;D:\Softwares\wildfly-<i>version</i>.Final.zip&lt;/entry&gt;
</pre>
You need to specify the absolute path of the ZIP file of JBoss WildFly application server. For simple installation, you need to specify the absolute path of the ZIP file of JBoss WildFly application server.</td></tr>
</tbody>
</table>

**Description of entries in the SILENT Install File**

```
<!--Common Config settings-->
<!--if commonConfig_useDefault is true then the common config directory will be-->
<!--based off the installationRoot setting above and the commonConfig_directory 
setting-->
<!--will be ignored. If commonConfig_useDefault is set to false then you must 
provide-->
<!--a valid directory location for the commonConfig_directory setting.-->
<entry key="commonConfig_useDefault">true</entry>
<entry key="commonConfig_directory">c:\tibco\MDMcommon</entry>
<entry key="tibco.cim.common.dir">c:\tibco\MDMcommon</entry>
```

The directory for common configuration (set to `true` by default) is based on the Install root. To change this, enter `false` for `CommonConfig_useDefault` and provide the new location in the `commonConfig_directory`.

The `tibco.cim.common.dir` is an additional key for external use of the common directory. This is used for post install processing. The value should be the same as of `commonConfig_directory` key.

```
<!--MDM XMLC library download Settings->
<entry key="LGPLAssemblyLicenseAccepted">true</entry>
<entry key="LGPLAssemblyDownload">true</entry>
<entry key="LGPLAssemblyPath">c:\tibco\thirdpartyDownload</entry>
```

The LGPL license is accepted (set to `true` by default). By default, LGPL assembly download option is set as `true`. If `LGPLAssemblyDownload` is `false`, then `LGPLAssemblyPath` contains already downloaded assembly files.

```
<!--MDM Configurator Settings-->
<entry key="httpPort">6080</entry>
<entry key="stopPort">6009</entry>
<entry key="startServer">false</entry>
```

Provide the `httpPort` and `stopPort` of the Configurator or keep the default as is. By default, the Configurator will not start after the installation is complete; to override this, set `startServer` to `true`.

```
<!--CIM Enterprise creation -->
<entry key="mdm.create.enterprise">true</entry>
<entry key="mdm.enterprise.name">demo</entry>
<entry key="mdm.admin.user.name">admin</entry>
<entry key="mdm.admin.password">Admin$12</entry>
<entry key="mdm.template.type">party</entry>
<entry key="mdm.import.data">true</entry>
```

For simple installation, you need to specify the enterprise details, administration credentials, and the template details.

- For `mdm.admin.password`: The password must contain eight alphanumeric characters with at least one lowercase and one uppercase alphabets and one numeric entry. Additionally, if you want to add special characters, change the default value of the `Password Required` property. For information on password policy, refer to the section, "Configuration Properties for Password" in *TIBCO MDM System Administration*. If password policy is not satisfied, the following error message is displayed: `SEC-5539 : Password must contain '8' alphanumeric characters with at least one lowercase and one uppercase alphabets and one numeric entry.`
- For `mdm.template.type`: specify the `customer`, `insurance`, `party`, or `none` option. The value is case-sensitive and must be in lowercase. If you specify `false` for the `mdm.create.enterprise`, specify `none` for the `mdm.template.type` and `false` for the `mdm.import.data`.

# Configurator

The Configurator is an independently deployable web-based configuration utility that allows you to configure various properties for TIBCO MDM. It can be used for the initial setup of the application as well as for ongoing maintenance of the TIBCO MDM configuration.

### Prerequisite

To start the Configurator on on-premises or enterprise servers, ensure that you have set the `MQ_NO_FS` environment variable to `false`.

Type the following URL into your browser:
*protocol*://*host*:*port*`/config/index.html`

For example, `http://localhost:6080/config/index.html`.

## Manually Configuring TIBCO MDM

If you have not specified configuration information during TIBCO MDM installation, Configurator does not start. You need to perform manual configuration to start Configurator. However, TIBCO recommends you to specify configuration information during TIBCO MDM installation.

### Procedure

1. Specify Connector Port
   a) Navigate to `$MQ_HOME/configurator/tomcat/conf` directory and open the `server.xml` file.
   b) Replace the value of Connector port attribute to `6080`. For example,

   ```
   <Connector port="${mdm.server.http.port}" >
   ```

   Change to

   ```
   <Connector port="6080" >
   ```

   c) Save the `server.xml` file.
2. Set *JRE_HOME*
   a) Navigate to `$MQ_HOME/configurator/tomcat/bin` and open the `setenv.bat` file.
   b) Specify the value for *JRE_HOME* parameter.
   c) Save the `setenv.bat` file.
3. Start Configurator
   a) Navigate to `$MQ_HOME/configurator/tomcat/bin` folder and run the `setenv.bat` file
   b) Navigate to `$MQ_HOME/configurator/tomcat/bin` and run the `startup.bat` file.

   The Configurator starts. You can verify the logs related to Configurator in the `$MQ_HOME/configurator/tomcat/logs` folder. For detailed information on the Configurator, see the "Configurator" chapter in *TIBCO MDM System Administration*.

## Configuring MDM to Connect with JMS

To connect to TIBCO EMS, enter the values for the Queue Setup and Bus Setup configuration properties.

**Prerequisites**

- For Queue Setup: Ensure that the Cluster Association is set to `TIBCOCluster` (Queue Setup > Queue Definition > Default Queue)
- For Bus Setup: Ensure that the Cluster Association is set to `TIBCOCluster` (Bus Setup > Topic Setup > Default Topic)

### Procedure

1. Log into Configurator.
2. Navigate to **InitialConfig** > **Queue Setup** > **Messaging Cluster** > **TIBCO EMS** .
3. Enter the values for the following properties:

   *Queue Setup Properties*

| Property Name | Description | Example Values |
|---|---|---|
| TIBCO EMS Server Connection URL | Enter the address and port for the property. | `tcp://10.97.118.57:7222` |
| `TIBCO EMS Cluster Name` | Clustering of the EMS server is supported by TIBCO MDM. Enter the values for these properties configured for each server. | `LocalhostServer` |
| `TIBCO EMS Server Default Encoding` | | `UTF-8` |

4. Navigate to **InitialConfig** > **Bus Setup** > **Cluster** > **TIBCO EMS** .
5. Enter the JMS server address and port for the Localhost Server Connection String property.
6. Click **Save** to save the entered values.

## Superuser Password

The TIBCO MDM application runs under the context of a UNIX or Windows user. This user account has to be created.

After you install a new instance of TIBCO MDM, the default superuser information is as follows:

On UNIX, it is recommended that the root user is not used to run MDM.

- Default company name:`tibcocim`
- User: `tadmin`
- Password: `euc!1dAl`

`Al` stands for `Alexander`, that is, capital `A` and small `l` letter.

Additionally, the new password configuration properties are added in Configurator. For more information, refer to the section, "Configuration Properties for Password" in *TIBCO MDM System Administration*. If you configure the `Password Required` property in Configurator to add special characters in the password, you need to customize the display error message ( SEC-5539) in the `UserText.properties` file. For customization, refer to the section "Localize Text Strings" in *TIBCO MDM Customization Guide*.

## Changing the Superuser Password

After logging in, you can change the password from UI using My Account Profile and Modify User screens.

### Procedure

1. Using My Account Profile Screen
   a) Click *User name-Company name* on the upper right corner of the main UI screen. The My Account Profile screen is displayed.
   b) Type the old password in the **Old Password** field.
   c) Type the new password in the **Password** field.
   d) Re-type the new password in the **Re-enter Password** field.
   e) Click **Save**. The new password is saved.
2. Using Modify User Screen
   a) Click **Administration** > **User Accounts** . The User Accounts screen is displayed.
   b) Select the check box next *tadmin* user name.
   c) Click **Modify**. The Modify User screen is displayed.
   d) Type the new password in the **Password** field.
   e) Re-type the new password in the **Re-enter Password** field.
   f) Click **Save**. The new password is saved.

# Configuration Requirements

The configuration requirements depend on if this is for a new installation, an upgrade installation, or if any new properties need to be set for a specific release.

For new installations, all TIBCO MDM configurations described in this topic must be set to meet your site’s standards.

For upgrade installations, TIBCO MDM configurations that were set during the original installation are automatically updated by the migration utility. Any new properties that need to be set for a specific release are listed in the TIBCO MDM Release Notes.

Any new properties that need to be set for a specific release are listed in the TIBCO MDM Release Notes.

## Setting Crontab

The `tibcocrontab.sh` sample script is in `$MQ_HOME/bin`. Use the script to create crontab entries required for a periodic cleanup of temporary files. The temporary files are generated in the `$MQ_COMMON_DIR/Work` and `$MQ_COMMON_DIR/Temp` directories.

You can modify the `tibcocrontab.sh` file to change the frequency and retention policy and to fit within your IT policies. If your IT policy does not allow a cron job to be setup, you can use any schedule. You can run this script manually or using any job scheduler.

If you are defining crontab entries, ensure the crontab is defined for a user that owns the TIBCO MDM Application.

**Prerequisite**

For Linux or Unix installations, you must set `$MQ_HOME` and `$MQ_COMMON_DIR` in tibcocrontab.sh.

### Procedure

1. At the Unix Shell prompt, enter:

```
$crontab -e
```

2. Set `$MQ_LOG` environment variable by adding the following line: **`LOG_FILE=$MQ_HOME/log/velocrontab.log`**
3. Edit the file to add the following line:

   ```
   1 0 */30 * * sh /opt/tibco/mdm/version/bin/tibcocrontab.sh >> /opt/tibco/mdm/
   version/log/tibcocrontab.log 2>&1
   ```

   indicates you to run the `/opt/tibco/mdm/`*`version`*`/bin/tibcocrontab.sh` shell script with the sh shell at 00:01 every 30 days, appending (adding) the standard and error output to the `/opt/tibco/mdm/`*`version`*`/log/tibcocrontab.log` file.

   ```
   0 1 */1 * * find MQ_COMMON_DIR/Work -type d -empty -mmin +23 -print0 | xargs -0
   rm -rf;
   ```

   indicates you to execute the find command on the `$MQ_COMMON_DIR/Work` directory. The command produces a empty sub-directories list in the `$MQ_COMMON_DIR/Work` that is at least 23 hours old. Piping that list (with a null terminator to accurately capture directory names with spaces and other non-standard characters) to the **`xargs`** command, so that it can build and execute the **`rm -rf <directory list>`** command (the most efficient way to remove empty sub-directories).
4. Save and exit.

   :wq!
5. Edit the `RETENSION_POLICY` variable in this script to specify the retention policy. `RETENSION_POLICY_DAYS=`*`numberofdays`*
6. To check any crontab setting, run the following command:

   ```
   $crontab –l
   ```

## JRE Configuration Requirements

TIBCO MDM uses Sun's JRE with WebLogic and IBM's JRE with WebSphere on all other platforms. You can set the configuration properties of TIBCO MDM using the Configurator.

The following code samples show the properties used for SSL setup. In order for the application server to function, one of these two properties must be set:

- Set SSL for Sun JRE: In the Configurator, set:
  - **Security Provider** > **Sun** > **SSL Protocol Handler Package** to com.sun.net.ssl.internal.www.protocol
  - **Security Provider** > **Sun** > **SSL Provider** to com.sun.net.ssl.internal.ssl.Provider.
- Set SSL for IBM JRE: In the Configurator, set:
  - **Security Provider** > **IBM** > **SSL Protocol Handler Package** to com.ibm.net.ssl.internal.www.protocol.
  - **Security Provider** > **IBM** > **SSL Provider** to com.ibm.jsse.JSSEProvider.

## Java Configuration Changes

Make the Java configuration changes listed in this topic.

1. Remove all JAR files from the `$JAVA_HOME/jre/lib/ext` directory.
2. Change the security provider list in the `$JAVA_HOME/jre/lib/security/java.security` file to:

   ```
   security.provider.1=com.ibm.crypto.provider.IBMJCE
   security.provider.2=com.ibm.jsse.JSSEProvider
   security.provider.3=sun.security.provider.Sun
   ```

## Specifying Database User Details

Specify the database user details specific to each database in Configurator.

### Procedure

1. Log in to Configurator.
2. Select the **Basic** configuration outline
3. Select **Database**. The common properties related to database are displayed in the Table List pane on the right side.
4. Enter the database user details in the following fields. The database values are based on the selected database in the **Settings** > **Database** option. The available database options are Oracle and SQL Server.

    - Database Name
    - Database User Name
    - Database Password

    For information on the database credentials, refer to

    - For Oracle: Creating Database Users for Seed Data Creation
    - For SQL Server: Creating Database User for Seed Data Creation
5. Click **Save** to save the database configuration changes.

### Result

For more information on the database configuration, refer to Setting up Databases.

## Database Analysis

It is highly recommended that you setup a periodic job for database analysis. The frequency of the job depends on data volumes, frequency of change, and your IT policies. Consult your DBA.

It is recommended that **analyze** be run whenever the number of rows in various tables is changed by more than 10%.

### Purge Historical Data

You can setup a periodic purge workflow to reduce the historical data.

## Setting Configuration for Decoupling of Documentation

TIBCO MDM documentation is decoupled from the product installation. Therefore, to integrate documentation in the application, select either `online` or `offline` for the `com.tibco.cim.help.url` (MDM Help Configuration) property in Configurator. The default value is `online`.

- **Online**: Using this mode, you can access the documentation online when you are in the network. In the `online` mode, by default the documentation is linked to the latest documentation site URL. For example, for the TIBCO MDM 9.1.0 release version, after clicking the Help icon in the application, the documentation is linked to `https://docs.tibco.com/pub/mdm/9.1.0/doc/html/index.htm` URL.
- **Offline**: Select this mode if you are off the network and want to access the documentation in the application. Stop the TIBCO MDM server if it is running, and then perform the following steps:

### Procedure

1. Copying Documentation Directory
   a) Go to https://docs.tibco.com and access TIBCO MDM documentation.
   b) Click the **Versions** tab to access the release specific documentation assembly.
   c) Click **Download All** to download all documentation including PDF and HTML Help.
   d) Save `tibco-mdm-`*`version`*`-documentation.zip` file to your local directory.
   e) Extract contents of the documentation ZIP directory.
   f) Go to `$MQ_HOME` and locate `ECM.ear\EML.war`.
   g) Create the doc sub-directory inside the help folder.
   h) Copy the html directory extracted from the documentation ZIP directory inside the doc sub-directory.
   i) Deploy the modified the ECM.ear file to the Application Server.
2. Specifying Configurations
   a) Restart the Configurator.
   b) Click **Advanced** in the **Configuration Outline** section. A list of advanced configuration categories is displayed.
   c) Select the **UI Setting** category.
   d) Select **offline** from the MDM Help Configuration property drop-down list.
   e) Click **Save**. The Save dialog box is displayed.
   f) Enter the description, if any.
   g) Click **Save**. The configuration changes are saved.
3. Hot Deploying on Application Server
   a) Go to **Initial Config***NodeID*
   b) Click **Save & Redeploy**. The Save & Redeploy dialog box is displayed. For Host Name, by default **localhost** is displayed.
   c) Enter the description, if any.
   d) Click **Save**. The MDM Help Configuration property is hot deployed on the application server.
4. Verifying Help
   a) Restart the TIBCO MDM server.
   b) Click the Help icon on any screen, the TIBCO MDM documentation is displayed.

# Uninstalling TIBCO MDM

The uninstaller presents panels using you can select choices about the product selection, product location, and so on.

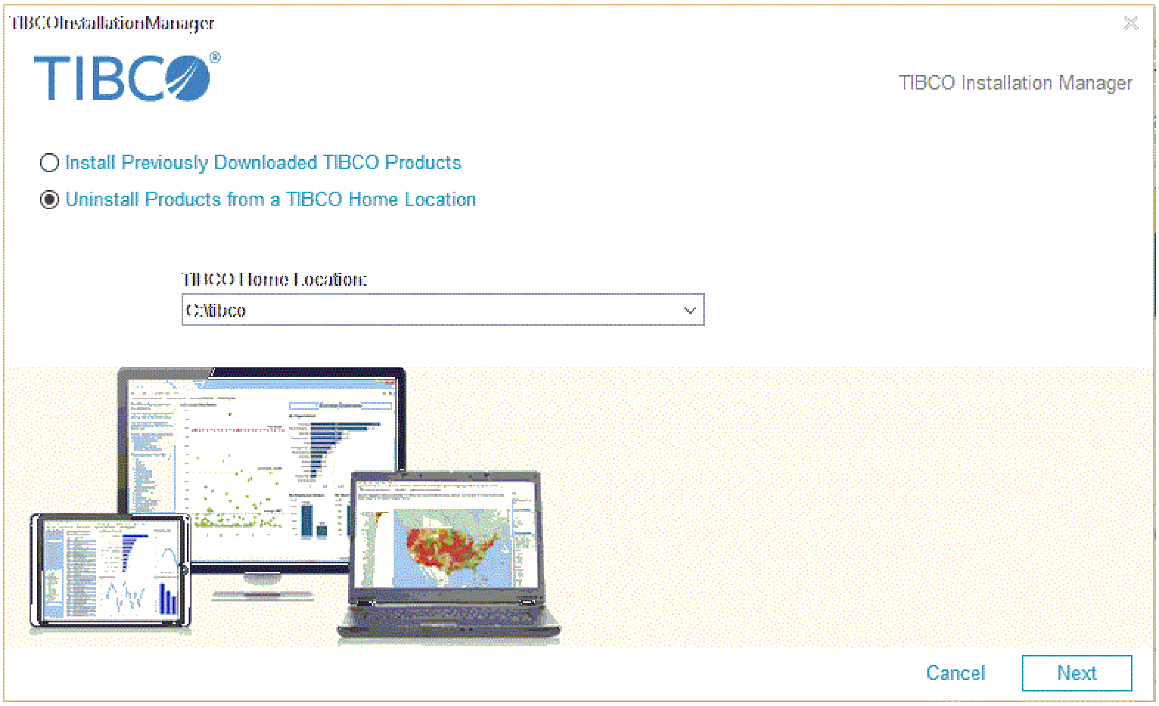
### Prerequisites

Before uninstalling TIBCO MDM, ensure that Configurator instances are not running (if they are when you attempt to uninstall TIBCO MDM, some files may be locked and uninstallation may not take place correctly).

### Procedure

1. On Windows: navigate to `$MQ_HOME/tools/universal_installer` and run `TIBCOUniversalInstaller-x86-64.exe` or navigate to **All Programs** > **TIBCO** and click **Uninstall**.
2. On UNIX, navigate to `$MQ_HOME/tools/universal_installer` and run `TIBCOUniversalInstaller-lnx-x86-64.bin`.

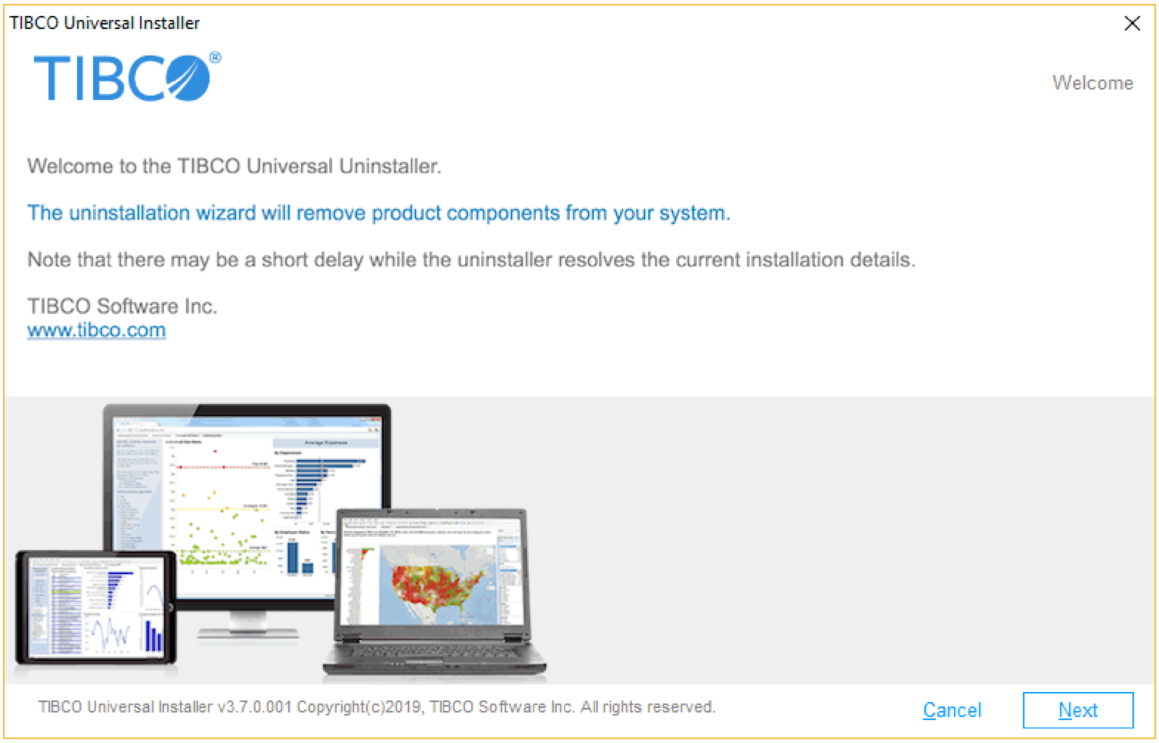
   The uninstaller tries to stop the Tomcat server for the Configurator. If the Tomcat server for the Configurator is already stopped, a Connection refused error is displayed and an exception is thrown as the server is already stopped. Ignore the error message and the exception.
3. In the TIBCOInstallation Manager window, select the **Uninstall Products from a TIBCO Home Location** option. The TIBCO home location is automatically displayed.

   ○ Install Previously Downloaded TIBCO Products
   ● Uninstall Products from a TIBCO Home Location

   TIBCO Home Location: C:\tibco

   Click **Next**.
4. In the Welcome window, review the information.



TIBCO Universal Installer

Welcome

Welcome to the TIBCO Universal Uninstaller.

The uninstallation wizard will remove product components from your system.

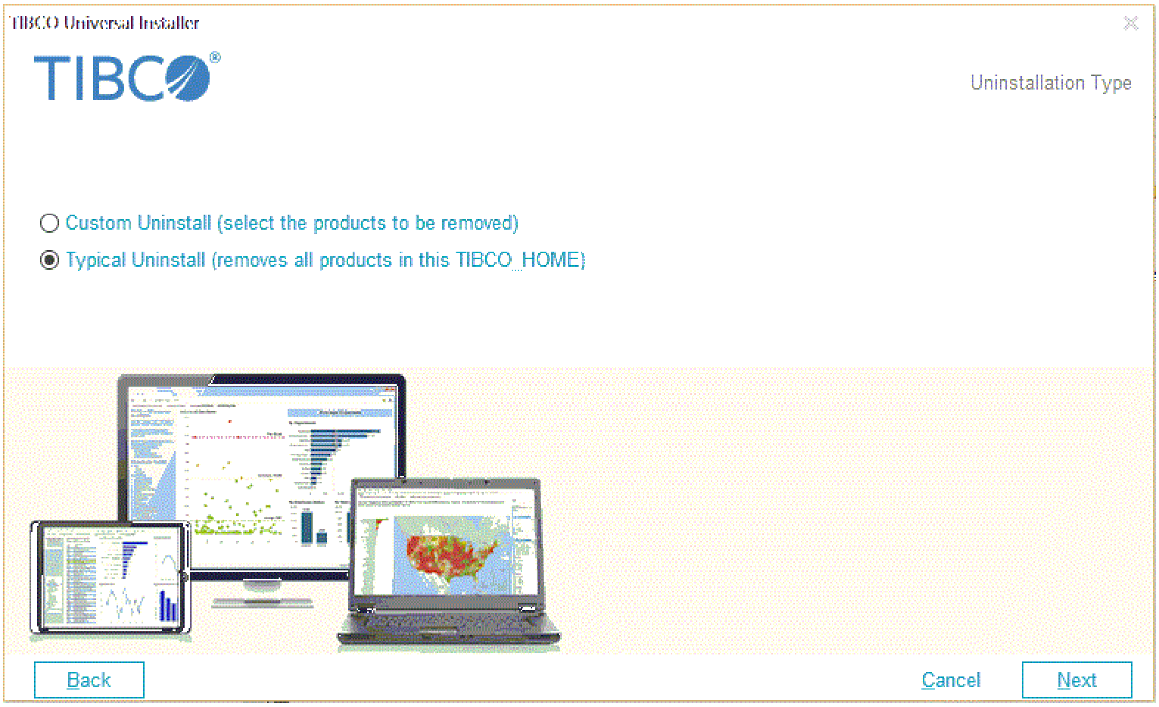
Note that there may be a short delay while the uninstaller resolves the current installation details.

TIBCO Software Inc.
www.tibco.com

TIBCO Universal Installer v3.7.0.001 Copyright(c)2019, TIBCO Software Inc. All rights reserved.

Click **Next**.

5. In the Uninstallation Type window, select the **Typical Uninstall (removes all products in this TIBCO_HOME)** option to uninstall all the products.



TIBCO Universal Installer

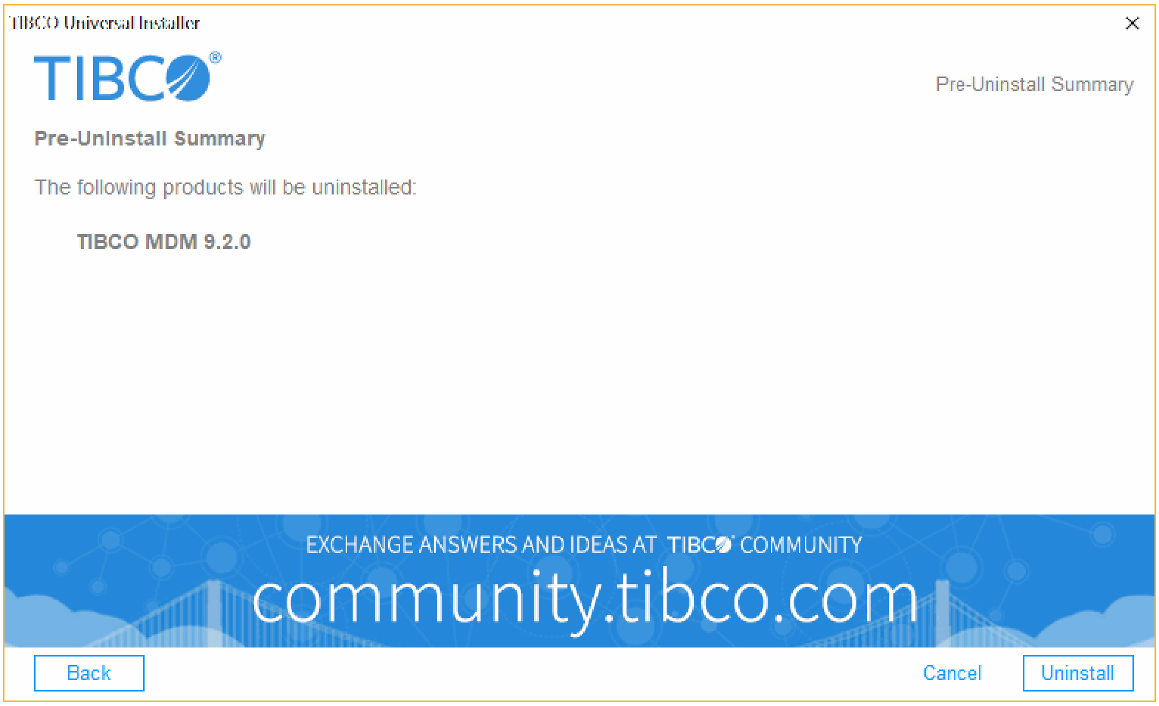
Uninstallation Type

○ Custom Uninstall (select the products to be removed)

● Typical Uninstall (removes all products in this TIBCO_HOME)

Click **Next** to continue with the uninstallation. The installer configures your uninstallation choices.

6. In the Pre-Uninstall Summary window, review the information and ensure that it is correct.


TIBCO Universal Installer
×
TIBCO
Pre-Uninstall Summary
EXCHANGE ANSWERS AND IDEAS AT TIBCO COMMUNITY
community.tibco.com
Back
Cancel
Uninstall

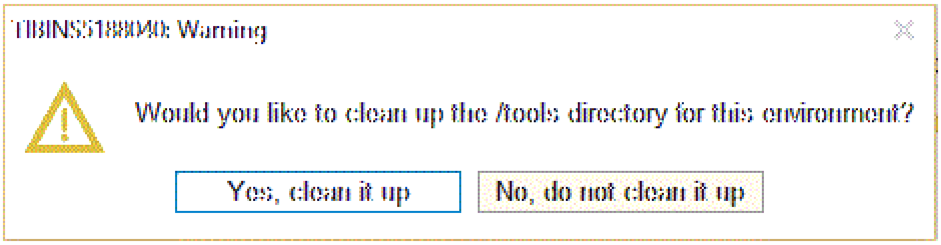
Pre-Uninstall Summary

The following products will be uninstalled:

TIBCO MDM 9.2.0

- If you want to change any of your choices, click **Back**. You can then restart the uninstallation process from the starting point.
- If you are satisfied with your choices, click **Uninstall**. The installer performs the necessary uninstallation tasks.

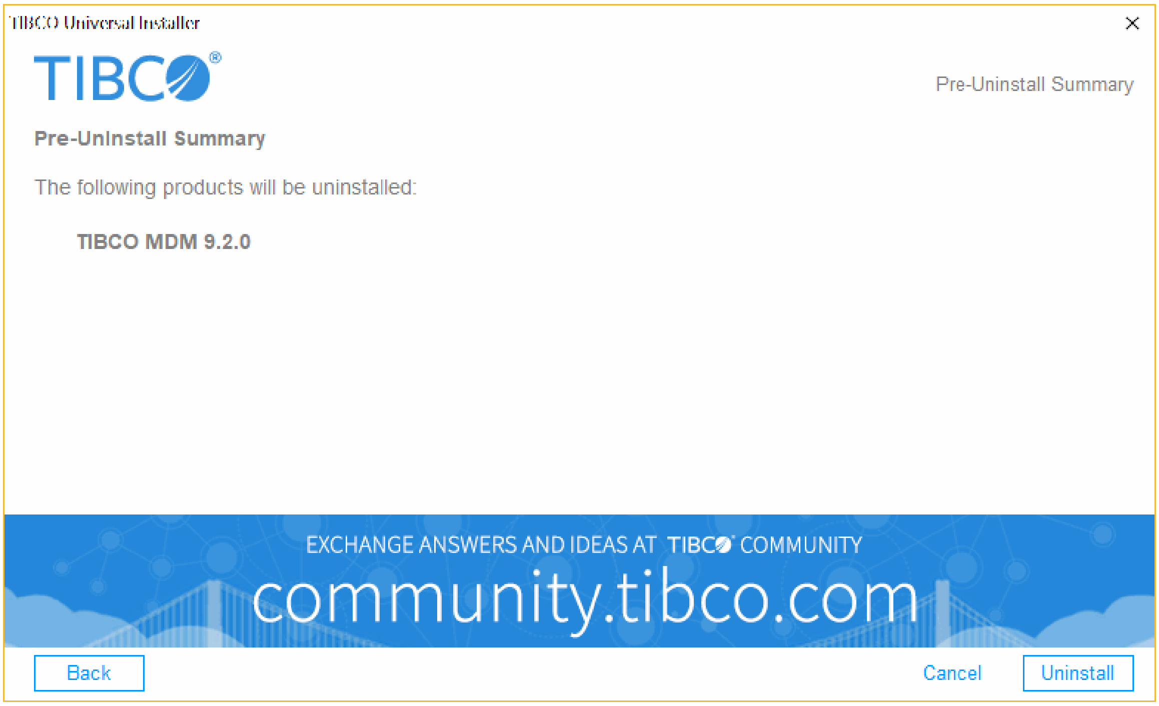
7. When uninstallation has completed, the clean up message is displayed.



Would you like to clean up the /tools directory for this environment?

Click the **Yes, clean it up**.

8. In the Post Uninstall Summary window, click **Finish** to exit from the installer.



TIBCO Universal Installer

Pre-Uninstall Summary

**Pre-Uninstall Summary**

The following products will be uninstalled:

**TIBCO MDM 9.2.0**

# Setting up Databases

TIBCO MDM supports Oracle and SQL Server databases for data storage.

Configuring the database of TIBCO MDM consists several steps, most of which are done manually such as creating users, schemas, and table spaces; changes to SQL scripts to change location or default table space sizes, handling multibyte characters, and so on.

## Supported Databases

Install anyone of the following databases as per your requirement. For a list of versions and platforms supported, refer to the *Readme.txt* file shipped with installation of TIBCO MDM.

*Supported Databases*

| Database | Description | Site Reference |
|---|---|---|
| Oracle | Download the supported version of the Oracle database from the Oracle site and unzip it to the required location. | http://www.oracle.com/technetwork/indexes/downloads/index.html?ssSourceSiteId=ocomen#menu-downloads<br>Install the Oracle server software as directed in the Oracle installation document. |
| SQL Server | Download the latest version of Microsoft SQL Server from the following site and unzip it to the required location | http://msdn.microsoft.com/en-us/sqlserver/bb671149.aspx<br>Install the SQL Server software as directed in the appropriate SQL installation document. |
| PostgreSQL | The PostgreSQL database is installed with the Simple installation. You do not need to separately download it. For more information on simple installation, refer to Simple Installation with PostgreSQL. | Not Applicable |

## Database Sizing Requirements

The database sizing requirements include the minimal sizing requirements for `init`*`dbname`*`.ora` `(initecm50.ora)`; for small, medium, and large databases.

*Database Sizing Requirements*

| Database Parameter Setting | Low-end | Mid-range | High-end |
|---|---|---|---|
| Db block size | 8192 | 8192 | 8192 |
| Db_file_multiblock_read_count | 8 | 16 | 32 |
| Shared pool size | 75 MB | 150 MB | 200 MB * |

| Database Parameter Setting | Low-end | Mid-range | High-end |
|---|---|---|---|
| Processes | (# of application servers)* (application server max db connection pool size) + 200 | | |
| Parallel max servers | 2 | 4 | 5* |
| Log_buffers | 25% of the system memory | | |
| Timed_statistics | True | True | True |
| Max_dump_file_size | 5 MB | 10 MB | 20 MB |
| Rollback_segments | 8 seg | 16 seg | 32 seg* |
| Open cursors | 300 | 450 | 3000 |
| Character set | UTF-8 | | |
| Buffer Pool Size | 150 MB | 300 MB | 500 MB - 2 GB* |
| db_writer_processes | 75% of the cpu_count parameter value | | |
| Sessions | 1.1* processes + 200 | | |
| optimizer_mode | ALL_ROWS | | |
| shared_servers | # of dispatchers* 2 | | |
| Transactions | # sessions | | |

* These values depend on various factors including concurrent users, message and workflow volumes, number of records, and so on. Your DBA should adjust these values based on the actual load and required performance characteristics.

## Configuration of Oracle Database

To configure the Oracle database, use the Oracle Configuration Assistant. Consult your Database Administrator on standard practices followed by your IT department to change the recommended structure according to your needs.

**Prerequisites**

- Ensure that all required environment variables are set. See Environment Variables.
- Ensure that Oracle Client Software Developer Edition or Enterprise Edition is installed on the computer hosting the application server ().

- Ensure that the **`sqlldr`** utility is available.
- Use the latest driver provided by Oracle.
- A valid and tested connect string should be present in the `TNSNAMES.ora` file. For example, the connection **`URL: jdbc:oracle:oci:@ORACLERAC`**

  where `ORACLERAC` is the TNS entry in the client's `TNSNAMES.ora` file and oci drivers are used to support TAF.

**TNSNAMES.ora file (client)**

```
ORACLERAC =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = hostname1.domainname.com)(PORT =
      1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = hostname2.domainname.com)(PORT =
      1521))
    )
    (FAILOVER=on)
    (LOAD_BALANCE = ON)
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = orcl)
      (FAILOVER_MODE =
        (TYPE = SELECT)
        (METHOD = BASIC)
        (RETRIES = 180)
        (DELAY = 5)
      )
    )
  )
```

The TNS entry supports both failover and load balancing.

- Database port
- Database server host name

## Recommended Configuration for TIBCO MDM

Follow the recommended configuration for setting up the database.

The following is the recommended configuration:

- Two sets of Redo logs with two members each.
- Analyze regularly (for instance, weekly) or after significant change in data (for example, after importing records in the database).
- Database to be run in ARCHIVE LOG MODE.
- Mirrored control files.
- Remove INSTALL user after the database is created.

On Linux 5.1, for Oracle > sqplus to work, disable SELINUX as follows:

```
echo 0 > /selinux/enforce
```

## Creating Database User for Oracle Database

You need to create a database user for seed data creation.

If you select **Use an Existing MDM Database User** option while setting up the Oracle database in Configurator, the database user should be available. For information on the Database Setup Wizard, refer to Installing Seed Data Using Database Setup Wizard.

### Procedure

1. Connect to the database.
2. Run the createuser.sql script from `$MQ_HOME/db/oracle/configure`.
3. Specify the database *username* and *password*.

   The database user is created.

## Creating Seed Data Manually for Oracle Database

You can create seed data by using the database setup wizard or through manual configuration. This is especially useful if you want to create the seed data independent of the database installation.

To install seed data with the database wizard, see Installing Seed Data Using Database Setup Wizard.

For a new database installation, you must create a database user ID and use the database installation scripts to create all database objects. For manual seed data creation, perform the following installation steps:

### Procedure

1. Modify `$MQ_HOME/db/oracle/install/install.bat/.sh` to change the *USER*, *PASSWORD*, *mdmInstanceName*, *mdmInstanceDesc*, and *INSTANCE* variables. The *USER* and *PASSWORD* variables should be set to the user that you created.

   > For non-Windows environment, ensure that the *PATH* variable includes the Bourne shell or a compatible shell before running scripts to create seed data.

2. From the install directory, run the script `install.bat/.sh`.

   The script creates the sequences, tables, indexes, triggers, and views in the appropriate table spaces and inserts seed data into the tables.

## Creating Table Spaces

Table spaces are required to hold data and indexes for all tables required for TIBCO MDM and for all data sources uploaded.

TIBCO MDM uses the following different kinds of tables and indexes:

- Fixed tables
- Indexes for fixed tables
- Data source tables
- Master catalog tables
- Indexes for master catalog tables

### Procedure

1. Change the default data file location specified in the createtablepsace.sql file based on your Oracle installation.
   a) Go to `$MQ_HOME/db/oracle/configure/`.
   b) Open the createtablepsace.sql file.
   c) Change the data file location for all table spaces. For example, if you have installed Oracle in the E:/app/oradata/orcl folder, change the `C:/oracle/`*ver*/oradata/orcl path to E:/app/oradata/orcl.
2. Run the createtablepsace.sql script to create the following table spaces:

*Tablespaces*

| Tablespace Name | Description |
|---|---|
| VELODBDATA | For fixed TIBCO MDM tables. |
| VELODBINDX1 | For fixed TIBCO MDM table indexes. |
| VELODBDF | For TIBCO MDM data source tables |
| VELODBDATA | For TIBCO MDM master catalog tables. |
| VELODBINDX | For TIBCO MDM master catalog table indexes. |

- The VELODBDATA1 table space has a minimum size of 100 MB and maximum size of 500 MB. The remaining table spaces are 100 MB. You can change the size of the table spaces based on you requirement. You can also set the size as unlimited for the table spaces.
- If you are migrating from the earlier versions of Oracle to Oracle 12c version, you have to explicitly grant UNLIMITED TABLESPACE to the user. Oracle has discontinued the support granting UNLIMITED TABLESPACE to the RESOURCE role user.

  Run the following command:
  GRANT UNLIMITED TABLESPACE TO *username*

3. For complete installation (table spaces and seed data), run the installation script from `$MQ_HOME/db/oracle/configure/doall`.bat or doall.sh
   a) To create table spaces without seed data, run `$MQ_HOME/db/oracle/configure/createusertablespace`.sh or .bat
   b) To create only seed data, see Creating Seed Data Manually.

## One Two or Three Table spaces Option

If you choose to create one, two, or three table spaces, you need to modify the sqlscripts in the `$MQ_HOME/db/oracle/install/scripts/ddl` directory to create data structures in the appropriate table spaces.

- One table space for data and indexes
- Two table spaces, one for data and one for indexes
- Separate table spaces for each set of tables
- A combination of one of these three options

*Database SQL Scripts - One, Two, or Three Tablespaces Option*

| SQL Script | Description |
|---|---|
| **create_tabs.sql** | All fixed tables used in TIBCO MDM are created by this script. These tables are created by default in the VELODBDATA1 table space. The name of the table space needs to be modified appropriately in this script. If the default storage parameters are not good enough for some tables, contact your Database Administrator to modify this script to create tables with the appropriate storage parameters. |
| **create_PK.sql** | The primary keys for all the fixed tables are created by this script. These primary keys are created by default in the VELODBINDX1 table space. The table space name needs to be modified appropriately in this script. |
| **create_indexes.sql** | The indexes for all the fixed tables are created by this script. These indexes are created by default in the VELODBINDX1 table space. The table space name needs to be modified appropriately in this script. |
| **create_ds_tables.sql** | All data source tables used in TIBCO MDM are created by this script. These tables are created by default in the VELODBDF table space. The name of the table space needs to be modified appropriately in this script. |

After modifying the SQL scripts, you need to perform the two steps required for the five table space option. The output of these scripts goes into the log file `$MQ_HOME/db/oracle/install/logs/output.log`. Ensure that there are no errors during the execution of these scripts.

## Creating Table spaces - Alternative Approach

In this approach, TIBCO MDM creates and manages table spaces. A separate table space is created for each enterprise that is used for data source upload.

This approach is not recommended.

### Procedure

1. Create two different table spaces, VELODBDATA and VELODBINDX.
2. Create mdmadm user identified by *password* default table space VELODBDATA temporary table space temp.
3. Grant connect, resource to mdmadm.
4. Alter user mdmadm quota 1000m on VELODBINDX.
5. Grant create tablespace privilege to mdmadm.
6. Grant UNLIMITED TABLESPACE privilege to mdmadm.
7. Set the following properties using Configurator:
   (Advanced view, Database, Oracle).
   a) Set DatabaseTablespace Create Privilege to True.

   For CSV data upload, you need to set the option to False.

b) Set DatabaseTable Space Name to VELODBDF. The default value is USERS.

c) Set DatabaseMaster Catalog/Repository Data Table Space to VELODBDATA. The default value is USERS.

d) Set DatabaseMaster Catalog/Repository Index Table Space to VELODBINDX. The default value is USERS.

> With this approach, you need to create table space privileges for the TIBCO MDM database user.

8. If TIBCO MDM is responsible for creating a table space for each enterprise on the first data load, set the following properties using Configurator (Go to **Initial Config** > **Database** *databasename* ):

   -
     ```
     Table Space File Size
     Table Space Next Size
     Table Space Initial Storage Size
     Table Space Increase %
     ```

     For the following properties, go to **Initial Config** > **Database**:

   -
     ```
     Dynamic Table Space Location
     Table Space Name Prefix
     ```

     After completion, you can create the seed data. See One Two or Three Table spaces Option.

## Deleting User and Table spaces

Follow these steps to remove the user and table spaces that were created with the installation.

> This removes all data related to TIBCO MDM and should be used very carefully.

### Procedure

1. Modify the `$MQ_HOME/db/oracle/configure/dropuser.sql` script to change the user name to the user you created. By default, this script drops user **mdmuser**.

2. Modify the `$MQ_HOME/db/oracle/configure/droptablepsace.sql` script to change the names of the table spaces to what you have created.

3. From the configure directory, run the script dropusertablespace.bat. This script removes the user and table spaces.

   > In some cases, the data files may not get removed. Delete them manually.

## Handling Multibyte Characters

When the Oracle database is created, the database charset is set to UTF-8. In addition to the UTF-8 charset, the TIBCO MDM database needs to be deployed with a configuration for character semantics that enable globalization and enhance code portability.

From Oracle9i onwards, you can set the default character semantics at either the session or instance level using the NLS_LENGTH_SEMANTICS parameter. This parameter must be effective before a table is created.

The NLS_LENGTH_SEMANTICS parameter decides how strings should be stored in the database, as characters or bytes. TIBCO recommends that strings in the database be stored as characters (CHAR). This setting takes care of globalization issues with strings.

- For a New TIBCO MDM Installation

  The database object creation scripts specify the correct semantics. If you use these scripts, you do not have to perform any additional steps.

- For an Existing TIBCO MDM Installation

Prior to TIBCO MDM 7.1, TIBCO MDM instances were created using the seed data export dump provided. Depending on the database NLS_LENGTH_SEMANTICS, the tables may not have been created with the correct semantics to handle NLS characters. You can convert the older instances to the correct semantics as follows:

To convert an existing schema from byte semantics to character semantics and a single-byte character set to a multibyte character set:

### Procedure

1. Export the schema.
2. Set the NLS_LENGTH_SEMANTICS parameter using either of the following methods:
   - In the init.ora file, set NLS_LENGTH_SEMANTICS=**CHAR**.
   - Issue the following command on the target database (This command modifies the NLS_INSTANCE_PARAMETERS and NLS_SESSION_PARAMETERS views. Storage is now measured in characters and not bytes):

     ALTER SYSTEM SET NLS_LENGTH_SEMANTICS=CHAR SCOPE=BOTH

     > This resolves defects associated with Data too long for the Column when multibyte characters are used.
3. Stop and restart the database so that the parameter change takes effect.
4. Drop the original schema.
5. Recreate the original schema and its tables. You can use the IMPORT SHOW=Y option to get the CREATE TABLE statements. Columns in the recreated tables will now use character semantics (as it is the default). The import command generates the create table statements.
6. Run the create table statements and the schema is created.
7. Import the schema into the target database using the "IGNORE=Y" IMPORT option.
8. Export the dump and save it.
9. Retest the dump as follows:

   alter system set nls_length_semantics=byte;
10. Import the dump.

### Result

The tables should have correct semantics even when NLS_LENGTH_SEMENTICS is set to byte.

## Limitation for Data Source Upload Using the sqlldr utility

A data source with many columns hangs in the upload stage for long time and later an error occurs. This is using the Oracle `sqlldr` utility. When large file is provided, the SQL loader needs memory to run. Java separates the process, which has its own overhead. Therefore, memory required is almost double of what it takes normally. The only solution is to reduce the file size or increase memory available on your computer.

## Troubleshooting with Oracle Database

Resolve the errors that you may come across while configuring the Oracle database.

*Troubleshooting with Oracle Database*

| Issue | Description | Solution |
|---|---|---|
| Bad Interpreter Issue | A "bad interpreter" error is displayed on UNIX. | The first line of all scripts on UNIX must be as follows:<br>`#!/usr/bin/sh`<br>Check whether or not the first line of the UNIX script follows this format. You can also create a soft link as follows:<br>ln -s /bin/sh /usr/bin/sh |
| Insufficient Shared Memory Issue | Oracle database error, unable to allocate required shared memory. (ORA-04031: unable to allocate x bytes of shared memory). | This error is related to the insufficient shared pool size allocation. Consult Oracle documentation and your DBA to resolve it. Usually the error may be resolved by increasing the amount of available shared memory. |
| Inserting and Updating Data from ProcessLog and ProcessState Tables | Two errors are intermittently thrown when inserting or updating data the ProcessLog and ProcessState tables, even though the data to be inserted has a valid value and length. This error has been observed on Oracle 10.1.0.2.0.<br>`ORA-01461: can bind a LONG value only for insert into a LONG column`<br>ORA-01483: invalid length for DATE or NUMBER bind variable | Restarting the application server may resolve the issue temporarily.<br>There are similar issues reported in Oracle MetaLink. Reference Document IDs: 241358.1, 461670.1<br>If the problem persists, contact Oracle support and consider upgrading to the latest patch. |

# Configuration of SQL Server Database

Install the SQL Server software as directed in the appropriate SQL installation document. Consult your Database Administrator about standard practices followed by your IT department to change the recommended structure according to your needs. After the SQL Server software is installed, configure the database.

TIBCO MDM works with SQL server configured with Windows authentication. For such setup, you need to configure JBoss WildFly application server. For information, refer to Configuring TIBCO MDM for JBoss WildFly Application Server.

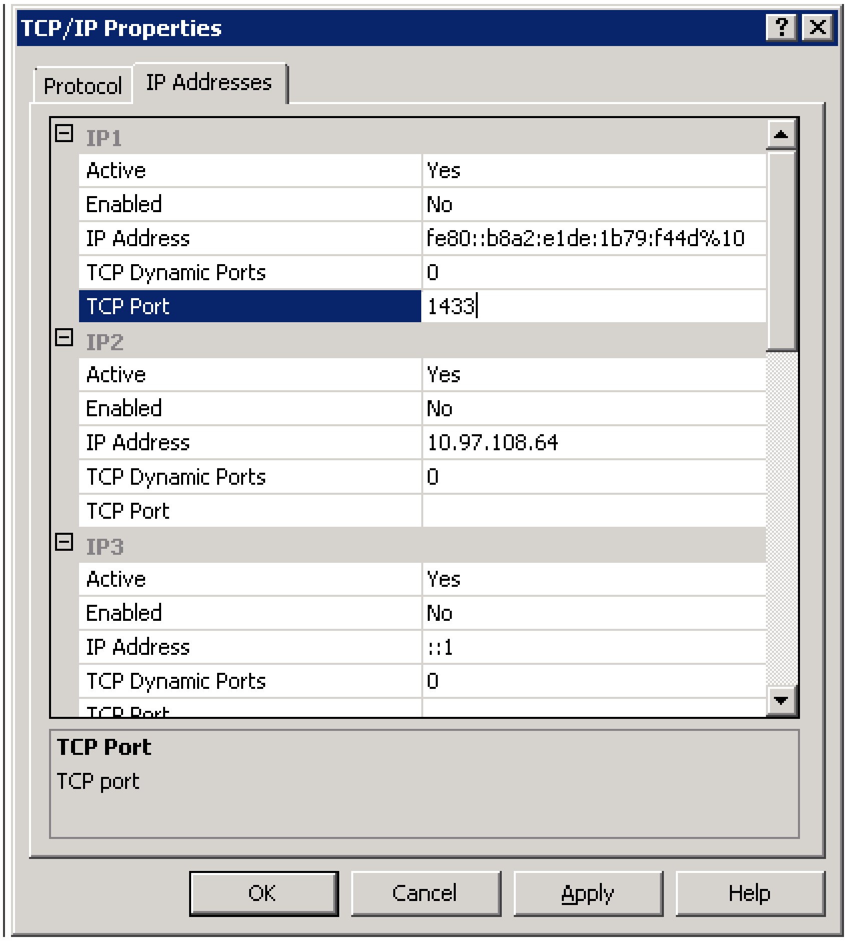
To improve the performance, the SQL Server database must be installed with collation as CaseInsensitive (CI).

## Setting TCP/IP Port

The SQL Server installation must use the fixed TCP/IP port.

### Procedure

1. Open SQL Server Configuration Manager.
2. In the left column, navigate to **SQL Server Network Configuration** > **Protocols**.
3. In the right column, in the **Protocol Name** field, double-click **TCP/IP**.
   The TCP/IP Properties window opens.



| | |
|---|---|
| **IP1** | |
| Active | Yes |
| Enabled | No |
| IP Address | fe80::b8a2:e1de:1b79:f44d%10 |
| TCP Dynamic Ports | 0 |
| TCP Port | 1433 |
| **IP2** | |
| Active | Yes |
| Enabled | No |
| IP Address | 10.97.108.64 |
| TCP Dynamic Ports | 0 |
| TCP Port | |
| **IP3** | |
| Active | Yes |
| Enabled | No |
| IP Address | ::1 |
| TCP Dynamic Ports | 0 |
| TCP Port | |

**TCP Port**
TCP port

4. Click the **IP Addresses** tab.
5. Set the **TCP Port** number to **1433**.
6. Click **OK**.

## Set Transaction Isolation Levels

You must set the transaction isolation levels to READ_COMMITTED_SNAPSHOT and ALLOW_SNAPSHOT_ISOLATION to avoid read locks in highly concurrent environment.

The snapshot isolation level specifies that data read within a transaction does not reflect changes made by another simultaneous transactions. The transaction uses the data row versions that exist when it begins. Therefore, when the data is read, no locks are placed and the snapshot transactions do not block other transactions from writing data. To enable the snapshot isolation levels, specify the following two properties:

- READ_COMMITTED_SNAPSHOT: Set the READ_COMMITTED_SNAPSHOT database option to ON to provide statement-level read consistency. The statements cannot read data values that are modified, but not yet committed by other transactions.

To set this parameter, log in as admin and run the following script with an appropriate database name:

```
ALTER DATABASE DATABASENAME SET READ_COMMITTED_SNAPSHOT ON
```

By default, the READ_COMMITTED_SNAPSHOT database option is set to OFF.

- ALLOW_SNAPSHOT_ISOLATION: Set the ALLOW_SNAPSHOT_ISOLATION database option to ON to provide transaction-level read consistency. If another transaction modifies the reading rows, the Microsoft SQL Server database engine instance retrieves the version of the row that existed at the start of the transaction. You can only use Snapshot isolation against a database.

  To set this parameter, log in as admin and run the following script with an appropriate database name:

  ```
  ALTER DATABASE DATABASENAME SET ALLOW_SNAPSHOT_ISOLATION ON
  ```

  By default, the ALLOW_SNAPSHOT_ISOLATION is set to OFF.

  For the READ_COMMITTED_SNAPSHOT and ALLOW_SNAPSHOT_ISOLATION levels, the read operations acquire only the Schema Stability (Sch-S) table level locks. It does not lock any pages or rows.

  These levels function similar to the SERIALIZABLE level, however you need to ensure that READ does not lock rows.

For more information to set these transaction isolation levels, refer to the Microsoft site (http://msdn.microsoft.com/en-us/library/ms173763.aspx).

## Creating Database User for SQL Server Database

You need to create database user for seed data creation.

If you select **Use an Existing MDM Database User** option while setting up the SQL Server database in Configurator, the database user should be available.

### Procedure

1. Login to the SQL Server Management Studio as a Super User.
2. Run the createUser.bat script from `$MQ_HOME/db/sqlserver/configure`.

   Where,

   *USERNAME* is the database user name and *PASSWORD* is password for the specified user name.

   The database user is created.

To provide permission to the user for creating database, do the following:

3. Open the SQL Server Management Studio, select *super username* **Security** > **Logins** .
4. Right-click *username* and select **Properties**. The Login Properties - *username* window is displayed.
5. In the left panel, select **Server Roles**. The server roles are displayed in the right panel.
6. In the Server Roles section, select the **sysadmin** check box.
7. Click **OK**.

   The system administration server role is created. This server role can grant server-wide security privileges to a user.

### Result

Also, specify the following parameters that are needed while installing TIBCO MDM:

- Database name

- Database port
- Database server host name or IP address

## Creating Seed Data Manually for SQL Server Database

You can create seed data by using the database setup wizard or through manual configuration.

To install seed data with the database wizard, see Installing Seed Data Using Database Setup Wizard.

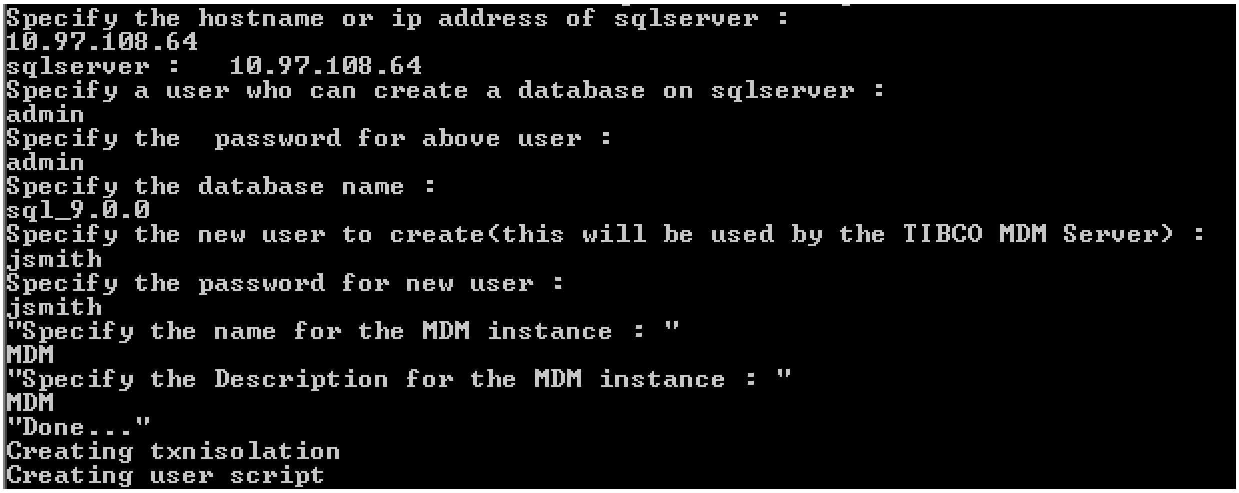
### Prerequisites

Set the *SQLSERVER_HOME* environment variable before running **`doall.bat`**, where the data files are created.

### Procedure

1. Open the command prompt and type the command to go to the SQL Server directory. For example: `C:\> cd %MQ_HOME%\db>` **`cd sqlserver`**
2. Open the Configure directory from the SQL Server location. For example: `C:\tibco\mdm\`*`version`*`\db\sqlserver>` **`cd configure`**
3. Run the `doall.bat` file. For example: `C:\tibco\mdm\`*`version`*`\db\sqlserver\configure>`**`doall.bat`**
   The script prompts for the following information:

   - IP address or host name of the SQL server
   - the user name and password of an administrator, who can create a database on SQL server
   - the database name that you want to create
   - the new user name that you can use in TIBCO MDM
   - the password of a new user
   - the name of MDM instance
   - the description for the MDM instance



```
Specify the hostname or ip address of sqlserver :
10.97.108.64
sqlserver :   10.97.108.64
Specify a user who can create a database on sqlserver :
admin
Specify the  password for above user :
admin
Specify the database name :
sql_9.0.0
Specify the new user to create(this will be used by the TIBCO MDM Server) :
jsmith
Specify the password for new user :
jsmith
"Specify the name for the MDM instance : "
MDM
"Specify the Description for the MDM instance : "
MDM
"Done..."
Creating txnisolation
Creating user script
```

4. Enter the details. The ddl and seed scripts start running. The scripts create the sequences, procedures, triggers, tables, and indexes, and inserts seed data into the tables. The log files are created in the `%MQ_HOME%\db\sqlserver\install\logs` folder.
5. Verify whether the following line is displayed in the console:

```
"Option is: isolation level Value: read committed snapshot"
```

After the doall.bat script is executed successfully, configure your application server for the newly created SQL database.

6. Alternative approach: To create seed data, run %MQ_HOME%\db\sqlserver\install\installbasic.bat

   You can verify the installed database in SQL Server Management Studio.

### Copy SQL Server Rules

After the seed data is created, run copyRulesForSqlServer.bat located in the $MQ_HOME/bin/migration folder to copy SQL Server-specific rules.

Ensure that you have specified the MQ_HOME and MQ_COMMON_DIR environment variables before running the copyRulesForSqlServer.bat file. For more information on environment variables, refer to Environment Variables.

## SQL Server Installation Verification

Verify whether or not SQL Server is properly installed, configured for TIBCO MDM, and running successfully.

Type the following sqlcmd statement on the command line:

```
sqlcmd -S SQL Server name -d database name -U mdm db user name -P mdm db user
password -q "Select NAME from ENTERPRISE where ID=0"
```

Ensure that the statement returns TIBCOCIM.

## Support for SQL Server Replication

The SQL Server database has a requirement to create a primary key on tables for replication. Some of the out-of-the-box tables that are provided with TIBCO MDM do not have primary key. To support SQL Server replication, CreateAdditionalPk.sql script is provided in the $MQ_HOME/db/sqlserver/utility folder. Run the CreateAdditionalPk.sql script to create a primary key on the out-of-the-box tables using existing columns.

### Limitations

- The script does not handle dynamically created tables.
- The script does not handle tables, which do not have sufficient columns to create a primary key. For such tables, add an identity column, and use that column as primary key.

## Troubleshooting with SQL Server Database

Resolve the errors that you may come across while configuring the SQL Server database.

*Troubleshooting with SQL Server Database*

| Issue | Description | Solution |
|---|---|---|
| Database Verification Message Issue | The Option is: isolation level Value: read committed snapshot message is not displayed in the console. | To resolve this issue, perform the following steps:<br>Navigate to `$MQ_HOME/db/sqlserver/configure`.<br>Open the txnisolation.SQL file and run the following commands in Microsoft SQL Server Management Studio:<br>`USE [master]`<br>`GO`<br>`ALTER DATABASE mdmuser SET`<br>`ALLOW_SNAPSHOT_ISOLATION ON`<br>`ALTER DATABASE mdmuser SET`<br>`READ_COMMITTED_SNAPSHOT on`<br>`GO`<br>`USE [mdmuser]`<br>`DBCC useroptions`<br>`GO`<br>After you run DBCC useroptions, isolation level is displayed as read committed snapshot. |

# Creating Seed Data Manually for PostgreSQL Database

You can create seed data through manual configuration.

### Procedure

1. Navigate to `$MQ_HOME\db\postgreSQL\install` and run the **`install.bat`** or **`install.sh`** script.
2. On the command line, enter the following details:

   - PostgreSQL home directory
   - IP address or host name and port of the PostgreSQL database. The default port for PostgreSQL is 5432. Ensure that it is not in use.
   - Database name that you want to create
   - Administrator credentials are required to create a database on PostgreSQL
   - The new user name that you can use in TIBCO MDM, the password of new user, and schema
   - Storage location for the data table space, index, data fragments table space, temporary data table space, temporary index table space for TIBCO MDM database
   - The name and description for TIBCO MDM instance

   The script creates the sequences, tables, indexes, triggers, and views in the appropriate table spaces and inserts seed data into the tables.
3. Verify the log file created in the `%MQ_HOME%\db\postgreSQL\install\logs` folder.
   The seed data for PostgreSQL database is created, you can use this as ...............

# Installing Seed Data Using Database Setup Wizard

The Database Setup Wizard of Configurator makes the database setup process easy and user-friendly.

The Database Setup Wizard is common for all three databases. However, some fields vary according to the selected database in the **Settings** > **Database** option. The available database options are Oracle and SQL Server. For more information on selecting the database option, refer to the Configurator chapter of *TIBCO MDM System Administration*.

**Prerequisites**

Before running the Database Setup Wizard, ensure the following:

- The database is installed.
- The database client is installed on the local computer.
- If a user has already been created, specify the schema credentials. If the user has not been created, specify the DBA credentials. For information, refer to Creating Database Users for Seed Data Creation.
- TIBCO MDM is installed and the environment variables are created.
- Database SQL scripts are available.
  - For Oracle: in `$MQ_HOME/db/oracle`.
  - For SQL Server: in `$MQ_HOME/db/sqlserver`.
- For Oracle, the table spaces are not created.
- The following database JDBC JAR files are copied in the `$MQ_HOME/configurator/tomcat/lib` folder for seed data creation:
  - For Oracle: `ojdbc8.jar` copy from `$ORACLE_HOME/jdbc/lib`
  - For SQL Server: `mssql-jdbc-7.2.2.jre11.jar` ( Microsoft JDBC Driver 7.2 for SQL Server, Java Development Kit (JDK) 11.0 ) download from Microsoft Download Center
- For Oracle Database client globalization support, the following JVM arguments have been added in the `$MQ_HOME/configurator/tomcat/bin/setenv.bat` file.
  - `-Duser.country=en`
  - `-Duser.language=en`

### Procedure

1. Log in to Configurator.
2. Click **Tools** > **Set up Database** . is displayed.
3. In the Database Setup Wizard for *databasename* with the Database Access Mode screen, select one of the following options:
   a) **Create New MDM Database User**: Select this option to create a new database user.
   b) **Use an Existing MDM Database User**: Select this option to specify details of an existing user.

   To specify details of a new database user or an existing database user, refer to the following table:

*New and Existing Database User Details*

| Field Name | Description |
|---|---|
| Database Host | The IP address or host name of the server where database is installed. |
| Database Port | By default, the port specific to each database is displayed. For example,<br>• For Oracle, the default database port is `1521`.<br>• For SQL Server, the default database port is `1433`.<br>You can change the port value, if required. |
| Database Name (TNS Name) | The name of the database where TIBCO MDM data should be installed.<br>For Oracle RAC, specify the SID of either of the two clusters of the RAC database. |
| DBA User Name | The user name of the database administrator. |
| DBA User Password | The password of the database administrator. |
| Test Connection | Click **Test Connection** to connect to the database and verify if the connection is successful. If the test connection is not successful, verify the specified database details. |
| New MDM Database User Name | The new user name used for the connection to the database. |
| New MDM Database User Password | The new password used for the connection to the database. |
| Confirm MDM Database User Password | Reenter the new password for confirmation. |
| **Note**: Remember the user name and password. | |

| Field Name | Description |
|---|---|
| Table Space Location<br>(For Oracle database) | The file system directory (absolute path) location where all table spaces are created. This must be a local directory on the computer where database is running. You can specify a custom location and provide a full path of the directory. The directory should have 'write permission' to write a file.<br>**Notes**:<br>• If you do not specify the table space location, by default the table spaces are created in the `$ORACLE_HOME/database` directory.<br>• It is recommended that the file system in this directory contains a significant amount of available space. This directory is assigned by a DBA and should be backed up on a regular basis. For example: /opt/oradata |
| Database File Location<br>(For SQL Server database) | The SQL Server database location. You can specify a custom location and provide a full path of the directory.<br>**Note**: Database File Location should have permission to write files. |
| Database File Location<br>(For SQL Server database) | The SQL Server database location. |
| Database Host | The IP address or host name of the server, where database is installed. |
| Database Port | By default, the port specific to each database is displayed. For example,<br>• For Oracle, the default database port is `1521`.<br>• For SQL Server, the default database port is `1433`.<br>You can change the port value, if required. |
| Database Name (TNS Name) | The name of the database where TIBCO MDM data is installed. |
| Database User Name | The user name used for the connection to the database. |
| Database User Password | The password used for the connection to the database. |

c) On the Database Details and Create New MDM Database User screens, click **Next**.

- If you are not a DBA user or do not have permission to create tablespace and a new user, you can create a table space and database user using the scripts. For information, refer to Configuration of Oracle Database and Configuration of SQL Server Database.
- For Oracle, if you select the **Use an Existing MDM Database User** option, refer to Creating Database Users for Seed Data Creation for the details of the existing database user.
- For SQL Server, if you select the **Use an Existing MDM Database User** option, refer to Creating Database User for Seed Data Creation for the details of the existing database user.

4. In the MDM Instance Details screen, enter the following details and click **Next**:

*MDM Instance Details*

| Field Name | Description |
|---|---|
| MDM Instance Name | Specify the instance name of TIBCO MDM. When you have multiple instances, and you want to install a database, you can specify a particular instance name. For example, preproduction and postproduction instances. An instance entry is added in the database table. |
| MDM Instance Description | Specify the instance description of TIBCO MDM. |

5. In the Storage Profile Details screen, select one of the following storage profile options:
   a) **Typical**: select this option to use the default values. A Typical profile installs table spaces for the Oracle database and the database file location for the SQL Server database.

      After you select the **Typical** storage profile option, the Confirm Storage Parameters screen is displayed. Confirm the default values and click **Install** to install the seed data. Refer to step 6.

   b) **Custom**: select this option to specify the customized values for the default table space.

      After you select the **Custom** storage profile option, the Custom Profile Setup screen is displayed. Specify the values, size, and location for the default table spaces.

   c) Click **Next**.

6. In the Confirm Storage Parameters screen, confirm the customized values. Click **Install** to install the seed data.

7. The MDM Seed Data Summary screen displays the success and error report of the seed data and schema creation.
   a) To view the schema and seed data log file, click **Open**.

   - For Oracle, by default, the log files are stored in `$MQ_HOME/db/oracle/install/logs` folder.

     You can ignore tablespaces errors. For example, `"ERROR:-ORA-01543: tablespace 'VELODBTEMP' already exists".`

   - For SQL Server, by default, the log files are stored in `$MQ_HOME/db/sqlserver/install/logs` folder.

8. Click **Finish** to complete the database setup process.

# TIBCO MDM Installation on Application Servers

The standard TIBCO MDM installer copies the file onto disk; however it does not do much configuration. You need to perform several configuration changes that are required in the Application Server.

You can install TIBCO MDM on the following Application Servers:

- JBoss WildFly or JBoss EAP Application Server: Download the executable from the following site and unzip it to the required location:

  `http://wildfly.org/downloads/` Refer to the documentation supplied with the WildFly Application Server for any additional instructions.

  To configure JBoss WildFly or JBoss EAP with TIBCO MDM, refer to Configuring TIBCO MDM for JBoss WildFly Application Server and Creating Subdirectory Structure.
- WebSphere Application Server: Ensure that WebSphere Application Server is installed. Refer to the appropriate installation documentation supplied with WebSphere Application Server for installation instructions.
- WebLogic Application Server: Ensure that WebLogic Application Server is installed. Refer to the appropriate installation documentation supplied with WebLogic Application Server for installation instructions.

To improve the performances of these application servers, see "Application Server Configuration" chapter in *TIBCO MDM Performance Tuning Guide*.

## Prerequisites for Application Servers

Before installing TIBCO MDM on application server, you need to verify various prerequisites.

- Ensure that TIBCO MDM is installed.
- Ensure that anyone of the following database that you have installed is up and running:
  - Oracle
  - SQL Server
  - PostgreSQL
- Ensure that Configurator is installed.
- Ensure that all required environment variables are set. See Environment Variables.
- Ensure that JMS is up and running.
- Ensure that seed data is created.
  - For PostgreSQL database, you cannot create seed data using Database Setup Wizard in Configurator. This is a limitation.

    Ensure that you have access to the Administrative console to install and configure TIBCO MDM. You can use the Administrative console for WebSphere and WebLogic application servers.
- Ensure the Java Versions

  Currently, TIBCO MDM requires the Java versions listed as follows. Consult the readme shipped with your installation of TIBCO MDM for the most up-to-date software requirements.
  - JBoss WildFly Application Server

    JRE 11 (Default)

  Sun JVM

- Weblogic Application Server

  JRE 8

  Sun JVM or JRrockit JVM

- WebSphere Application Server

  JRE 8

  IBM JVM

  For HP platforms, use HP JVMs.

TIBCO MDM is not certified with Open JDK. However, if you use Open Java and encounter TIBCO MDM problems that require support, download and point to the Oracle release (`JAVA_HOME`). You can then verify that the issue is reproducible before contacting TIBCO support.

# TIBCO MDM Installation on JBoss WildFly Application Server

### Prerequisite

Follow these instructions to install TIBCO MDM on JBoss WildFly or JBoss EAP Application Server. Ensure that JBoss support is enabled for TIBCO MDM using the `Application Server Name` property in the Configurator ***Node ID*** > **Application Server** > **JBOSS**.

## Directory Structure

The JBoss WildFly Application Server directory structure is different from its previous versions. Therefore, you should understand its directory structure. Navigate to the location where you have downloaded the JBoss Application Server WildFly verison.

The following table lists and describes each folder included in the `wildfly-`*`version`*`.Final` folder:

*wildfly-version Directory*

| Sub folder Name | Description |
|---|---|
| `appclient` | Contains configuration files, deployment content, and writable areas used by the application client container run from this installation. |
| `bin` | Contains start up scripts and configuration files. It also contains various command line utilities such as vault, add-user, and Java diagnostic report available only for Unix and Windows environments. |
| `bin > client` | Contains a client JAR file for use by non-maven based clients and a readme file. |
| `docs > schema` | Contains XML schema definition files. |
| `domain` | Contains configuration files, deployment content, and writable areas used by the domain mode processes run from this installation. |
| `modules` | Contains various modules that are used in the JBoss Application Server. JBoss Application Server WildFly version is based on a modular class loading architecture. |
| `standalone` | Contains configuration files, deployment content, and writable areas used by the single standalone server run from this installation. For more information, refer to Standalone Directory Structure. |

| Sub folder Name | Description |
|---|---|
| `welcome-content` | Contains default Welcome Page content. |

The following two modes are used to run the server:

- Standalone
- Domain

## Standalone Directory Structure

In the standalone mode, each JBoss Application Server WildFly *version* instance is an independent process similar to the previous JBoss versions, such as 3, 4, 5, 6, or 7.

*Standalone Directory Structure*

| Sub folder Name | Description |
|---|---|
| `configuration` | Contains configuration files of the standalone server that runs the installation. This is the single place for configuration modifications of the standalone server. |
| `data` | Contains the information written by server. It can be used while restarting of the server. |
| `deployments` | Includes end user deployment content for automatic detection.<br>**Note**: The server's management API is recommended for installing deployment content. File system based deployment scanning capabilities remain for developer convenience. |
| `lib > ext` | Contains JAR libraries referenced by application using the Extension-List mechanism. |
| `log` | Contains standalone server log files. |
| `tmp` | Contains temporary files generated by the server. |
| `tmp > auth` | Contains authentication tokens that are used to exchange with local clients. |

## Creating Subdirectory Structure

For creating a module, you need to create the sub-directory structure.

> Create any one of the following directory structure specific to the database that you have installed:
>
> - For JBoss WildFly version, the directory structure is `$JBOSS_HOME\modules\system\layers\base\com`.
> - For JBoss EAP 7.0 version, the directory structure is `$JBOSS_HOME\modules\system\layers\base\com`.

### Procedure

1. For Oracle database, create the hierarchal folders in the `$JBOSS_HOME\modules\system\layers\base\com` directory as follows:

- `oracle > ojdbc8 > main`

2. For SQL Server database, create the hierarchal folders in the `$JBOSS_HOME\modules\system\layers\base\com` directory as follows:

   - `microsoft > sqlserver > main`

3. For PostgreSQL database, create the hierarchal folders in the `$JBOSS_HOME\modules\system\layers\base\org` directory as follows:

   - `postgresql > main`

   After creating the sub-directory structure, perform the following steps:

   - Place the `module.xml` file specific to each database in the sub-directory. For information on creating a `module.xml` file for each database, see Module Creation.
   - Place the JAR files specific to each database in the sub-directory. For information on downloading or copying JAR files for each database, see the JDBC Related Libraries section in "Third Party Libraries".

## Module Creation

Class loading in JBoss Application Server - WildFly version is different than the previous versions of the JBoss Application Server.

The following are the major features of class loading in JBoss Application Server - WildFly version:

- based on the modules and need to define explicit dependencies on other modules.
- deployments in the modules do not have access to classes that are defined in JARs, unless an explicit dependency on those classes is defined.
- the deployers within the server implicitly add some commonly used module dependencies to the deployment, such as, the `javax.api` and `sun.jdk`. In this way, the classes become visible to the deployment at runtime.
- for some classes, the modules must be specified explicitly in the `MANIFEST.MF` file as dependencies or Class-Path entries. Otherwise, you may see `ClassNotFoundExceptions`, `NoClassDefFoundErrors`, or `ClassCastExceptions`.

After creating the hierarchal folders as mentioned in the Creating Subdirectory Structure, create the `module.xml` file for each database. Define the actual JAR file inside it, which contain the database driver. For example, if you have installed Oracle database, create a `module.xml` file in the `$JBOSS_HOME\modules\system\layers\base\com\oracle\ojdbc8\main` folder. For creating a `module.xml` file for each database, refer to the following samples:

### For Oracle database

```
<?xml version="1.0" encoding="UTF-8"?>
<module xmlns="urn:jboss:module:1.3" name="com.oracle.ojdbc8">
    <resources>
        <resource-root path="ojdbc8.jar"/>
        <!-- Insert resources here -->
    </resources>
    <dependencies>
                <module name="javax.api"/>
                <module name="javax.transaction.api"/>
    </dependencies>
</module>
```

### For SQL Server database

```
<?xml version="1.0" encoding="UTF-8"?>
<module xmlns="urn:jboss:module:1.3" name="com.microsoft.sqlserver">
    <resources>
```

```
        <resource-root path="sqljdbc42.jar"/>
    </resources>
    <dependencies>
        <module name="javax.api"/>
        <module name="javax.transaction.api"/>
        <module name="javax.xml.bind.api"/>
    </dependencies>
</module>
```

### For PostgreSQL database

```
<?xml version="1.0" encoding="UTF-8"?>
  <module xmlns="urn:jboss:module:1.3" name="org.postgresql">
       <resources>
           <resource-root path="postgresql-42.2.5.jar"/>
       </resources>
       <dependencies>
               <module name="javax.api"/>
               <module name="javax.transaction.api"/>
           </dependencies>
   </module>
```

The following tables describes the elements that are used in the `module.xml` file.

### Elements and Description of Module.xml File

| Element Name | Description |
|---|---|
| Module Name | It must match with the directory structure that you have created for each database. For example, the directory structure for Oracle database is `$JBOSS_HOME\modules\system\layers\base\com\oracle\ojdbc8`. Therefore, the module name is `com.oracle.ojdbc8`. |
| resource-root path | Specify the driver JAR file name based on the database that you have installed. The path is relative and default to the main directory. For example, if you have installed the PostgreSQL database, specify `<resource-root path="postgresql-42.2.5.jar"/>`. |
| Dependencies | Define any dependency. For example, all JDBC data sources are dependent on the Java JDBC API's. They are defined in the `javax.api` module, which is located at `modules/system/layers/base/javax/api/main` folder. |

## Configuring TIBCO MDM for JBoss WildFly Application Server

Some element tags in the `standalone.xml` file differ in the JBoss WildFly version and JBoss 7.1 and JBoss EAP 7.0 versions. For example, `<subsystem xmlns="urn:jboss:domain:transactions:1.4">` is changed to `<subsystem xmlns="urn:jboss:domain:transactions:3.0">`.

Ensure the correct tagging during configuration.

### Procedure

Perform the following actions in the `standalone.xml` file if you are using JBoss WildFly Application Server:

1. Enable access to remote server
   a) Open the `standalone.xml` file located in the `$JBOSS_HOME/standalone/configuration` directory.
   b) Change the value of an interface attribute from **management** to **public** in the following property:

```
<socket-binding name="management-http" interface="public" port="$
{jboss.management.http.port:9990}"/>
```

2. Specify system properties
   a) Add <system-properties> element after the <extensions> element.
   b) Under <system-properties> element, type the system property name for the name attribute and its value for the value attribute.

```
<system-properties>
<property name="MQ_HOME" value="C:/Apps/tibco/mdm/version"/>
<property name="MQ_CONFIG_FILE" value="C:/Apps/tibco/mdm/version/config/
ConfigValues.xml"/>
<property name="MQ_COMMON_DIR" value="C:/Apps/tibco/mdm/version/common"/>
<property name="MQ_LOG" value="C:/Apps/tibco/mdm/version/log"/>
<property name="NODE_ID" value="Member1"/>
<property name="PATH" value="C:/Apps/tibco/mdm/version/bin/as/2.1/bin;C:/Apps/
tibco/
mdm/version/bin/as/2.1/lib;${PATH}"/>
```

The `PATH` system property is applicable only for Windows environment.

```
<property name="LD_LIBRARY_PATH" value=" /Apps/tibco/mdm/version/bin/as/2.1/
bin:/Apps/tibco/mdm/version/bin/
as/2.1/lib:${ LD_LIBRARY_PATH}"/>
```

The `LD_LIBRARY_PATH` system property is applicable only for non-Windows environment.

```
<property name="ORACLE_HOME" value="C:/Apps/Oracle/product/11.2.0/dbhome_1"/>
<property name="log4j.ignoreTCL" value="true"/>
<property name="org.apache.tomcat.util.http.Parameters.MAX_COUNT"
value="5000"/>
<property name="org.apache.catalina.connector.URI_ENCODING" value="UTF-8"/>
<property
name="org.apache.catalina.connector.USE_BODY_ENCODING_FOR_QUERY_STRING"
value="true"/>
</system-properties>
```

The path separator must contain forward slash instead of a backward slash. For example, for `MQ_COMMON_DIR` - `C:/Apps/tibco/mdm/`*`version`*`/common`.

The following table describes property names and their description:

*Environment Variables for JBoss WildFly Application Server*

| Property Name | Description |
|---|---|
| `MQ_HOME` | Refers to `$MQ_HOME` of TIBCO MDM. |
| `MQ_CONFIG_FILE` | Refers to configuration directory location of TIBCO MDM. |
| `MQ_COMMON_DIR` | Refers to common directory location of TIBCO MDM. |

| Property Name | Description |
|---|---|
| `MQ_LOG` | Refers to the log folder location specified in `$MQ_HOME`. |
| `NODE_ID` | Refers to the node ID. |
| `PATH` | Refers to the path to include ActiveSpaces libraries for Windows environment. |
| `LD_LIBRARY_PATH` | Refers to the path to include ActiveSpaces libraries for non-Windows environment. |
| `ORACLE_HOME` | Refers to the path where Oracle database is installed. |
| `log4j.ignoreTCL` | To configure the logging, specify True value for this property. After you configure this property, all logs are displayed in the respective log file. |
| `org.apache.tomcat.util.http.Parameters.MAX_COUNT` | By default, JBoss WildFly version application server set the `512` value for HTTP parameters. To override this value and allow maximum HTTP parameters, set the maximum value. For example, `5000` . |
| `org.apache.catalina.connector.URI_ENCODING` | Refers to the UTF-8 encoding. This needs to be specified to support multiple languages. |
| `org.apache.catalina.connector.USE_BODY_ENCODING_FOR_QUERY_STRING` | Specify `true`. The valid values are `true` or `false`. |

3. Specify **`max-post-size`** and **`max-parameters`**:

| Parameter Name | Description | Example |
|---|---|---|
| **`max-post-size`** | For uploading a file either through a web service or the TIBCO MDM UI, the maximum file size limit is 10 MB. If you want to upload a file greater than 10MB, add the **`max-post-size`** parameter and change the file size. You also need to change the value of the `Upload File Size Limit` property in Configurator.<br>For information on uploading a file through the TIBCO MDM UI, refer to the section, "Creating Records" in *TIBCO MDM User's Guide*. | `<server name="default-server">`<br>`<http-listener name="default" socket-binding="http" redirect-socket="https" enable-http2="true" max-post-size="974247881"/>`<br>`<https-listener name="https" socket-binding="https" security-realm="ApplicationRealm" enable-http2="true"/>`<br>`<host name="default-host" alias="localhost">`<br>`<location name="/" handler="welcome-content"/>`<br>`<filter-ref name="server-header"/>`<br>`<filter-ref name="x-powered-by-header"/>`<br>`</host>`<br>`</server>` |

| Parameter Name | Description | Example |
|---|---|---|
| **`max-parameters`** | The maximum number of parameters that can be added to the TIBCO MDM URL into the browser. Using this value, you can avoid the hash exposure used in the URL. This applies to both query and POST data parameters.<br>By default, the size of the **`max-parameters`** is `1000`. You can change the value as per your requirement. | `<server name="default-server">`<br>`<http-listener name="default" socket-binding="http" redirect-socket="https" enable-http2="true" max-post-size="974247881" max-parameters="5000"/>`<br>`<https-listener name="https" socket-binding="https" security-realm="ApplicationRealm" enable-http2="true"/>`<br>`<host name="default-host" alias="localhost">`<br>`<location name="/" handler="welcome-content"/>`<br>`<filter-ref name="server-header"/>`<br>`<filter-ref name="x-powered-by-header"/>`<br>`</host>`<br>`</server>` |

4. Add the following VM parameters in `standalone.conf` (Linux) or `standalone.conf.bat` (Windows):

```
--add-exports=java.base/jdk.internal.misc=ALL-UNNAMED
--add-exports=java.base/sun.nio.ch=ALL-UNNAMED
--add-exports=java.management/com.sun.jmx.mbeanserver=ALL-UNNAMED
--add-exports=jdk.internal.jvmstat/sun.jvmstat.monitor=ALL-UNNAMED
--add-exports=java.base/sun.reflect.generics.reflectiveObjects=ALL-UNNAMED
--illegal-access=permit
--add-opens jdk.management/com.sun.management.internal=ALL-UNNAMED
```

For example, in `standalone.conf.bat` (Windows) set parameters as shown in the following sample:

```
 "JAVA_OPTS=%JAVA_OPTS% --add-exports=java.base/jdk.internal.misc=ALL-UNNAMED"
"JAVA_OPTS=%JAVA_OPTS% --add-exports=java.base/sun.nio.ch=ALL-UNNAMED"
"JAVA_OPTS=%JAVA_OPTS% --add-exports=java.management/com.sun.jmx.mbeanserver=ALL-
UNNAMED"
"JAVA_OPTS=%JAVA_OPTS% --add-exports=jdk.internal.jvmstat/
sun.jvmstat.monitor=ALL-UNNAMED"
"JAVA_OPTS=%JAVA_OPTS% --add-exports=java.base/
sun.reflect.generics.reflectiveObjects=ALL-UNNAMED"
"JAVA_OPTS=%JAVA_OPTS% --illegal-access=permit"
"JAVA_OPTS=%JAVA_OPTS% --add-opens jdk.management/
com.sun.management.internal=ALL-UNNAMED"
```

5. Create data sources

a) Under <datasources> element, add <DataSource> element with attributes such as:

- `jndi-name="java:jboss/eCMDataSource"`
- `pool-name="MDMDataSource"`

a) Enable data source

- Specify `true` for the enabled attribute

a) Under <drivers> element add JDBC driver specific to each database.

- For Oracle database:

```
<driver>oracle.ojdbc.driver.OracleDriver</driver>
```

- For SQL Server database:

  ```
  <driver>com.microsoft.sqlserver.jdbc.SQLServerDriver</driver>
  ```

- For PostgreSQL database:

  ```
  <driver module="org.postgresql.jdbc.driver" name="PostgresqlDriver">
          <driver-class>org.postgresql.Driver</driver-class>
  </driver>
  ```

a) Specify connection settings:

- Connection URL-Type any one of the following URLs:

  For Oracle database: `jdbc:oracle:thin:@localhost:1521:`*dbinstance_name*

  For SQL Server database: `jdbc:sqlserver://localhost:1433;databaseName=`*value*

  For PostgreSQL database: `jdbc:postgresql://localhost:5432/`*dbinstance_name*

- User Name and Password

  Specify credentials for the Username and Password attributes.

a) Configure Transaction Isolation

- Specify the `TRANSACTION_READ_COMMITTED` value for <transaction-isolation> attribute.

a) Specify pool size

By default, `0` pool size is displayed.

- Specify 10 for Min Pool Size: `<min-pool-size>10</min-pool-size>`
- Specify 150 for Max Pool Size: `<max-pool-size>150</max-pool-size>`

a) Specify transaction timeout

- Specify the transaction timeout in the `<blocking-timeout-millis>` element. This element indicates the maximum time in milliseconds to block a transaction while waiting for a connection and before displaying an exception. This blocks only while waiting for a permit for a connection, and does not display an exception if creating a new connection that takes an inordinately long time. The default is **36000** milliseconds.

For sample data sources, refer to Sample Data Sources.

6. Change deployment timeout

   By default, the deployment timeout is displayed as `60` seconds. For slower machines, TIBCO recommends to increase the deployment timeout:

   a) Under `<subsystem xmlns="urn:jboss:domain:deployment-scanner:2.0">`element, add the deployment-timeout attribute and its value. For example,

   ```
    <deployment-scanner path="deployments" relative-to="jboss.server.base.dir"
   scan-interval="5000" runtime-failure-causes-rollback="$
   {jboss.deployment.scanner.rollback.on.failure:false}" deployment-
   timeout="5000"/>
   ```

7. Specify default timeout

   To deploy huge metadata from TIBCO MDM Studio to TIBCO MDM, you need to add the default timeout attribute under transaction element.

   a) Under `<subsystem xmlns="urn:jboss:domain:transactions:3.0">` element, add the following tag:

   ```
   <coordinator-environment default-timeout="1800000"/>
   ```

8. Change HTTP Port

   This step is optional. By default, the HTTP port is `8080`. If required, you can change it.

a) Under the `<socket-binding-group>` element, change the value of the port attribute for http port.

> Alternatively, you can also change the HTTP port using the following parameter: `standalone.bat -Djboss.socket.binding.port-offset=`*new _port_number*. For example, if the current HTTP port is 8080 and you want to change it to 8180, type `standalone.bat -Djboss.socket.binding.port-offset=`*100*;in the command prompt and press Enter. The HTTP port number is changed to 8180.

9. Configuration for EJB

- Passing Values by Reference

  While invoking the remote method of EJB, JBoss WildFly Application Server WildFly passes default values using the Passed By Value method instead of the Pass By Reference method. Therefore, to disable the Passed By Value method, add the following property in the `standalone.xml` file.

a) Navigate to `<subsystem xmlns="urn:jboss:domain:ejb3:4.0">` section, and add the following line at the end of the section:
`<in-vm-remote-interface-invocation pass-by-value="false"/>`

- Define New Pool and Increase EJB Pool Size

  When multiple users execute concurrent requests, multiple instances of each bean are used concurrently. Each bean has a limited number of instances in the bean-instance pool, available for use. If all the beans are in use, subsequent requests have to wait for a bean to be released by the previous thread, and then release back into the pool. This wait is for a specific time, that is, five minutes in TIBCO MDM. If the bean is not released within five minutes, an `Instance-Acquisition-TimeOut` error is displayed and the request remains incomplete.

  To avoid such errors, increase the EJB pool size to a large number (500 or 1000). On the JBoss WildFly Application Server, the default pool size is `20`, defined in the `standalone.xml` file. Changing the default pool size may affect all beans in all deployed applications. Therefore, you need to define a new pool.

a) Navigate to the `<subsystem xmlns="urn:jboss:domain:ejb3:4.0">` section, and define the following new pool and specify its pool size:

```
<pools>
<bean-instance-pools>
<strict-max-pool name="mdm-pool" max-pool-size="500" instance-acquisition-
timeout="5"          instance-acquisition-timeout-unit="MINUTES"/>
</bean-instance-pools>
</pools>
```

10. Defining TIBCO module

In some cases, additional external modules need to be created to configure out-of-the-box functionality.

a) Navigate to the `$JBOSS_HOME/modules/system/layers/base/com` folder and create the following hierarchical folders: `tibco/mdm/main`

b) Create a `module.xml` file in the `$JBOSS_HOME/modules/system/layers/base/com/tibco/mdm/main` folder and add the following snippets:

```
<module xmlns="urn:jboss:module:1.3" name="com.tibco.mdm">
    <resources>
        <resource-root path="tibcrypt.jar"/>
        <resource-root path="tibjms.jar"/>
        <!-- Insert resources here -->
    </resources>
<dependencies>
        <!-- Insert dependencies here -->
        <module name="javax.api"/>
        <module name="javax.jms.api"/>
    </dependencies>
</module>
```

c) Save the `module.xml` file.

11. Adding TIBCO Module to Global Modules List

> Adding TIBCO module to the Global modules list is must.

a) Open the `standalone.xml` file located at `$JBOSS_HOME/standalone/configuration` directory.

b) Modify the existing `<subsystem xmlns="urn:jboss:domain:ee:4.0"/>` section per database as follows:

- For Oracle and PostgreSQL databases:

```
        <subsystem xmlns="urn:jboss:domain:ee:4.0">
           <global-modules>
             <module name="com.tibco.mdm" slot="main"/>
           </global-modules>
        </subsystem>
```

- For Microsoft SQL Server database, you also need to add the Microsoft SQL driver module to the global Modules list. For example,

```
        <subsystem xmlns="urn:jboss:domain:ee:4.0">
           <global-modules>
             <module name="com.tibco.mdm" slot="main"/>
             <module name="com.microsoft.sqlserver" slot="main"/>
           </global-modules>
        </subsystem>
```

12. Specifying EMS configuration

a) Copy the `tibjms.jar` and `tibcrypt.jar` files from `$EMS_HOME/lib` and place them in the following folders:

*$JBOSS_HOME*/modules/system/layers/base/com/tibco/mdm/main *$MQ_HOME*/lib/external

b) Rename the `jboss-jms-api_2.0_spec-1.0.0.Final.jar` file to `jboss-jms-api_2.0_spec-1.0.0.Final.jar.org` located in the *$JBOSS_HOME*/modules/system/layers/base/javax/jms/api/main folder.

c) Copy the `jms-2.0.jar` file from `$EMS_HOME/lib` to the *$JBOSS_HOME*/modules/system/layers/base/javax/jms/api/main folder.

d) Open the `module.xml` file from the *$JBOSS_HOME*/modules/system/layers/base/javax/jms/api/main folder.

e) Replace the `path` value from `jboss-jms-api_2.0_spec-1.0.0.Final.jar` to `jms-2.0.jar`.

f) Restart the JBoss WildFly application server.

13. Specifying HTTP session timeout

The JBoss WildFly application server provides the default HTTP session timeout of 30 minutes. However, the JBoss Application Server does not support modification of the default value.

As per your business requirement, if you want to override the default HTTP session timeout value for TIBCO MDM, add the following entry in the application `web.xml` file or the `jboss-web.xml` file:

```
<web-app>
  <session-config>
   <!-- HTTP Session timeout, in minutes -->
   <session-timeout>40</session-timeout>
  </session-config>
</web-app>
```

Both the files are available in the `ECM.ear > EML.war > WEB-INF` directory.

## Sample Data Sources

You can create data sources for Oracle, SQL Server, and PostgreSQL databases. Refer to the sample data sources while configuring TIBCO MDM with JBoss WildFly application server. The sample data sources are available in the `standalone.xml` file.

### Sample data source for the Oracle database

```
...
...
<datasources>
...
...
   <datasource jndi-name="java:jboss/eCMDataSource" pool-name="MDMDATAsource"
jta="true" enabled="true" use-ccm="true">
     <connection-url>jdbc:oracle:thin:@localhost:1521:orcl</connection-url>
     <driver>OracleDriver</driver>
     <transaction-isolation>TRANSACTION_READ_COMMITTED</transaction-isolation>
             <pool>
                   <min-pool-size>10</min-pool-size>
                               <max-pool-size>150</max-pool-size>
     </pool>
     <security>
                                           <user-name>mdmuser</user-name>
                                           <password>mdmpassword</password>
     </security>
     <validation>
                                           <valid-connection-checker class-
name="org.jboss.jca.adapters.jdbc.extensions.oracle.OracleValidConnectionChecker"></
valid-connection-checker>
                                           <stale-connection-checker class-
name="org.jboss.jca.adapters.jdbc.extensions.oracle.OracleStaleConnectionChecker"></
stale-connection-checker>
                                           <exception-sorter class-
name="org.jboss.jca.adapters.jdbc.extensions.oracle.OracleExceptionSorter"></
exception-sorter>
     </validation>
             <timeout>
                                           <blocking-timeout-millis>72000</blocking-
timeout-millis>
             </timeout>
       </datasource>
       <drivers>
        ...
        ...
                                           <driver name="OracleDriver"
module="com.oracle.ojdbc8">
                                           <driver-class>oracle.jdbc.OracleDriver</
driver-class>
                                                   </driver>
        ...
        ...
       </drivers>
</datasources>
...
...
```

To establish the connection validator work with JBoss WildFly and Oracle, you must modify the `module.xml` file to add `com.oracle.ojdbc8` in the dependencies list. The file is located at *$JBOSS_HOME*`\modules\system\layers\base\org\jboss\ironjacamar\jdbcadapters\main`.

### Sample data source for the SQL Server database

```
...
...
<datasources>
...
...
```

```
       <datasource jndi-name="java:jboss/eCMDataSource" pool-name="MDMDATAsource" 
jta="true" enabled="true" use-ccm="true">
             <connection-url>jdbc:sqlserver://localhost:1433;databaseName=velodb</
connection-url>
             <driver>SQLServerDriver</driver>
     <transaction-isolation>TRANSACTION_READ_COMMITTED</transaction-isolation>
             <pool>
                   <min-pool-size>10</min-pool-size>
                   <max-pool-size>150</max-pool-size>
             </pool>
             <security>
                   <user-name>mdmuser</user-name>
                   <password>mdmpassword</password>
                     </security>
             <validation>
       <valid-connection-checker class-
name="org.jboss.jca.adapters.jdbc.extensions.mssql.MSSQLValidConnectionChecker"></
valid-connection-checker>
             </validation>
             <timeout>
                   <blocking-timeout-millis>72000</blocking-timeout-millis>
             </timeout>
           </datasource>
           <drivers>
        ...
        ...
                           <driver name="SQLServerDriver" 
module="com.microsoft.sqlserver">                               <driver-
class>com.microsoft.sqlserver.jdbc.SQLServerDriver</driver-class>
               </driver>
        ...
        ...
           </drivers>
</datasources>
...
...
```

### Sample data source for the PostgreSQL database

```
...
...
<datasources>
...
...
       <datasource jndi-name="java:jboss/eCMDataSource" pool-name="MDMDATAsource" 
jta="true" enabled="true" use-ccm="true">
     <connection-url>jdbc:postgresql://localhost:5432/velodb</connection-url>
             <driver>PostgresqlDriver</driver>             <transaction-
isolation>TRANSACTION_READ_COMMITTED</transaction-isolation>
             <pool>
                               <min-pool-size>10</min-pool-size>
                               <max-pool-size>150</max-pool-size>
             </pool>
             <security>
                               <user-name>mdmuser</user-name>
                               <password>mdmpassword</password>
             </security>
             <validation>
                               <valid-connection-checker class-
name="org.jboss.jca.adapters.jdbc.extensions.postgres.PostgreSQLValidConnectionCheck
er"></valid-connection-checker>
                               <exception-sorter class-
name="org.jboss.jca.adapters.jdbc.extensions.postgres.PostgreSQLExceptionSorter"></
exception-sorter>
     </validation>
             <timeout>
                               <blocking-timeout-millis>72000</blocking-timeout-
millis>
             </timeout>
           </datasource>
           <drivers>
```

```
        ...
        ...
                                   <driver name="PostgresqlDriver"
module="org.postgresql">
                                       <driver-class>org.postgresql.Driver</driver-
class>
                                   </driver>
        ...
        ...
           </drivers>
</datasources>
...
...
```

## Enabling Remote JMX Monitoring on JBoss WildFly Application Server

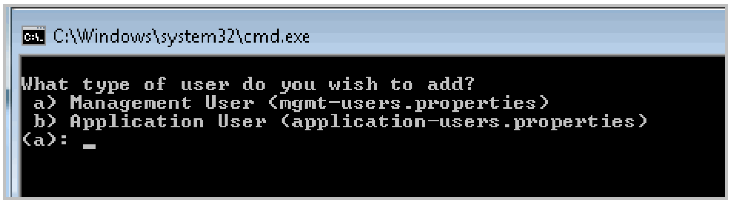
Perform the following steps for remote JMX monitoring using Java VisualVM for JBoss WildFly application server.

### Procedure

1. Create Management User Credentials

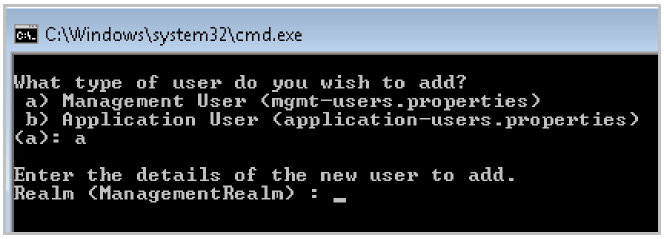
   While connecting to JavaVisualVM, you need to mention the management user credentials. Therefore, first create the management user credentials.

   a) Navigate to`$JBOSS_HOME/bin` directory and run `add-user.bat` or `add-user.sh` file. The following two options are displayed in the command prompt.



```
What type of user do you wish to add?
 a) Management User (mgmt-users.properties)
 b) Application User (application-users.properties)
(a): _
```

   b) Type a. You need to enter Realm credentials.



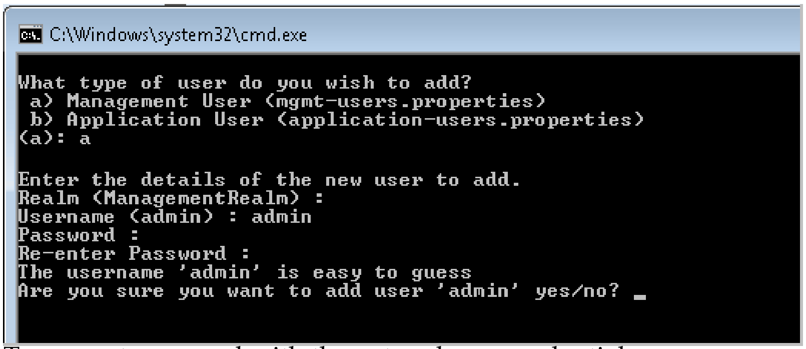
```
What type of user do you wish to add?
 a) Management User (mgmt-users.properties)
 b) Application User (application-users.properties)
(a): a

Enter the details of the new user to add.
Realm (ManagementRealm) : _
```

   Do not enter any values. Leave it blank.

   c) Press Enter. You need to enter user credentials.

   User name and password must be different, else an error is displayed.

   d) Type the user name, password, and confirm password.



C:\Windows\system32\cmd.exe

```
What type of user do you wish to add?
 a) Management User (mgmt-users.properties)
 b) Application User (application-users.properties)
(a): a

Enter the details of the new user to add.
Realm (ManagementRealm) :
Username (admin) : admin
Password :
Re-enter Password :
The username 'admin' is easy to guess
Are you sure you want to add user 'admin' yes/no? _
```

   e) Type `yes` to proceed with the entered user credentials.

      Management user is successfully created.

2. Modify the `standalone.xml` file
   a) Navigate to `$JBOSS_HOME/standalone/configuration` directory and open the `standalone.xml` file.
   b) Modify the `standalone.xml` file as follows:

```
<subsystem xmlns="urn:jboss:domain:jmx:1.3">
   <show-model value="true"/>
   <remoting-connector use-management-endpoint="true"/>
</subsystem>
```

   c) Save the `standalone.xml` file.

3. Start JBoss WildFly application server
   a) On the command line, run the script that includes environment variables.
   b) Navigate to `$JBOSS_HOME/bin`.
   c) Enter the following command:

```
./standalone.sh -Djboss.bind.address.management=IP address
```

4. Start Java VisualVM
   a) Navigate to `$JBOSS_HOME/bin/client` directory and copy `jboss-cli-client.jar` and place it to the `$JAVA_HOME/lib` directory.

      Ensure that `JAVA_HOME` refers to the location from where you want to run Java VisualVM.
   b) On the command line, type `$JAVA_HOME/bin`.
   c) Enter the following command:

```
jvisualvm --cp:a $JAVA_HOME\lib\jboss-cli-client.jar
```

      The JavaVisualVM window is displayed with the Local and Remote options.

5. Add JMX Connection
   a) For the Remote option, double-click or right-click to add the host name. After adding the host name, you need to add the JMX connection. Type the following URL in the Connection field:

      `service:jmx:remoting-jmx://`*`IPAddress`*`:9999`

      For example, `service:jmx:remoting-jmx://10.97.108.72:9999`.

      In this case, `9999` is the JMX port for the TIBCO MDM Server on 10.97.108.72.
   b) Specify the user credentials that you have created in Create Management User Credentials.
   c) Click **Connect**.

      If the JMX connection is successful, the connection URL link is displayed following the host name.

## Email Configuration

Configure email for the JBoss WildFly version.

Specify the following parameters in the `standalone.xml` file under the <socket-binding-group> section:

```
<socket-binding-group name="standard-sockets" default-interface="public" port-
offset="${jboss.socket.binding.port-offset:0}">
```

```
   <outbound-socket-binding name="mail-smtp">
   <remote-destination host="smtphostIP" port="25"/>
   </outbound-socket-binding>
</socket-binding-group>
```

- The value of the host attribute refers to the SMTP server.
- The values specified in Configurator for the SMTP Host and SMTP Port properties are not used. Rest email properties are used. For information on the email properties, refer to *TIBCO MDM System Administration*.

## Configuring JBoss WildFly for G11N Support

If you have JBoss WildFly application server and Microsoft SQL Server combination, you need to configure the `standalone.bat` file for G11N support.
For information on G11N support, refer to the section, "Globalization (G11n) Support" in *TIBCO MDM System Administration*.

### Procedure

1. Navigate to `$JBOSS_HOME/bin` directory.
   - For Windows: open the `standalone.bat` file.
   - For Linux: open the `standalone.sh` file.
2. Add the following snippet to the beginning of the file:
   For Windows:
   ```
   set "JAVA_OPTS=%JAVA_OPTS% -Dfile.encoding=UTF-8"
   ```
   For Linux:
   ```
   JAVA_OPTS="$JAVA_OPTS -Dfile.encoding=UTF-8"
   ```
3. Save the updated file.
4. Restart the server.

## Encrypting Password for Data Source

To encrypt the password for data source on JBoss WildFly application server, update the `standalone.xml` file.

### Procedure

1. Add SecureIdentityLoginModule and Security Domain
   a) Go to `$JBOSS_HOME/standalone/configuration` directory and open the `standalone.xml` file.
   b) Under <security-domains> element, add the **EncryptedPassword** security-domain and SecureIdentityLoginModule along with username and password module options. For example,
   ```
   <security-domains>
   <security-domain name="EncryptedPassword">
                       <authentication>
   <login-module
   code="org.picketbox.datasource.security.SecureIdentityLoginModule"
   flag="required">
                               <module-option name="username"
   value="@MDM.DB.USER@"/>
                               <module-option name="password"
   value="@MDM.DB.PASSWORD.ENTCRYPT@"/>
   <module-option name="managedConnectionFactoryName"
   value="jboss.jca:service=LocalTxCM,name=dataSource-1-PoolName,dataSource-2-
   PoolName"/>
                           </login-module>
                       </authentication>
   ```

```
                </security-domain>
</security-domains>
```

c) Under the <datasources> element, remove the following existing <user-name> and <password> from the <security> element because you have already added <user-name> and <password> as module options in the previous step:

```
<datasources>
<datasource ... >
.....
 <security>
  <user-name>mdmuser</user-name>
  <password>mdmpassword</password>
 </security>
</datasource>
</datasources>
```

d) Under the <datasources> element, add the **EncryptedPassword** security-domain, such as:

```
<datasources>
<datasource ... >
.....
<security>
     <security-domain>EncryptedPassword</security-domain>
</security>
</datasource>
</datasources>
```

e) Save the updated `standalone.xml` file.

2. Encrypt Password

a) On the command line, type `$JBOSS_HOME`.

b) Enter the following command:

```
java -cp modules\org\picketbox\main\picketbox-version.final.jar;modules\org
\jboss\logging\main\jboss-logging-version.GA.jar
org.picketbox.datasource.security.SecureIdentityLoginModule password
```

> Verify the versions of the picketbox and logging JAR files in the `$JBOSS_HOME` directory and specify them in the command accordingly.

The command returns the encrypted password. You can use the returned password in your security domain.

## Removal of jaxrs Entries for JBoss WildFly Application Server

The JBoss WildFly Application Server version provides RESTEasy framework to build RESTful web services and Java applications. It is implemented based on the JAX-RS specification. It conflicts with other RESTful framework, which are compliant with the JAX-RS specification.

Remove the jaxrs entries from the `standalone.xml` file located in the `$JBOSS_HOME/standalone/configuration` directory.

Before modifying the file, back up the existing `standalone.xml` file, and then remove the following lines:

```
<extension module="org.jboss.as.jaxrs"/>
<subsystem xmlns="urn:jboss:domain:jaxrs:1.0"/>
```

## Enabling SSL on JBoss WildFly Application Server

Enable and verify SSL on JBoss WildFly Application Server.

By default, JBoss WildFly application server uses `8443` port for the HTTPS protocol. Type the following URL into your browser: `https://IPaddress:8443/eml/Login`.

By default, the application realm is mentioned in the `standalone.xml` file located in the `$JBOSS_HOME/standalone/configuration` directory. The `application.keystore` is auto generated on the first use with a self-signed certificate for *localhost*. However, generating the self-signed certificates are discouraged because they result in browser warnings on internal sites.

It is always good practice to create a custom application realm. For creating a custom application realm on JBoss WildFly application server, see WildFly documentation.

## Deploying Custom Pages

To deploy the custom pages, you must configure the `standalone.xml` file.

### Procedure

1. Go to `$JBOSS_HOME/standalone/configuration` directory and open the `standalone.xml` file.
2. To deploy custom pages to a common folder in the cluster environment, add the following system property:

```
<system-properties><system-properties>
<property name="MQ_CUSTOMFORMS_HOME" value="C:/MDM_Installations/
MDMversionWFYORA_COMMON"/>
```

The configuration properties vary according to your application server. For deploying the custom pages in the cluster environment on WebLogic Application Server, see Configuring TIBCO MDM for WebLogic Application Server and for deploying the custom pages in the cluster environmenton WebSphere Application Server, see Configuring TIBCO MDM for WebSphere Application Server.

3. Add the location and path of the `MDMCustomui-content` file

```
<subsystem xmlns="urn:jboss:domain:undertow:3.1">
            <buffer-cache name="default"/>
            <server name="default-server">
                <http-listener name="default" socket-binding="http" redirect-
socket="https" enable-http2="true"/>
                <https-listener name="https" socket-binding="https" security-
realm="ApplicationRealm" enable-http2="true"/>
                <host name="default-host" alias="localhost">
                    <location name="/" handler="welcome-content"/>
                    <location name="/eml/components" handler="MDMCustomui-
content"/>
                    <filter-ref name="server-header"/>
                    <filter-ref name="x-powered-by-header"/>
                </host>
            </server>
            <servlet-container name="default">
                <jsp-config/>
                <websockets/>
            </servlet-container>
            <handlers>
                <file name="welcome-content" path="${jboss.home.dir}/welcome-
content"/>
                <file name="MDMCustomui-content" path="${MQ_HOME}/dynservices/
customui" follow-symlink="true"/>
            </handlers>
            <!--For the custom pages deployment in the cluster environment-->
             <handlers>
                <file name="MDMCustomui-content" path="${MQ_CUSTOMFORMS_HOME}/
dynservices/customui" follow-symlink="true"/>
            <filters>
```

```
                <response-header name="server-header" header-name="Server"
header-value="WildFly/10"/>
                <response-header name="x-powered-by-header" header-name="X-
Powered-By" header-value="Undertow/1"/>
            </filters>
        </subsystem>
```

4. Save the `standalone.xml` file.
5. Copy the existing custom page artifacts to the new location.

   ```
   cp -avr /home/apps/MDM_Installations/MDMversionJBOSSEAPSQL/mdm/version/
   dynservices/customui/home/apps/MDM_Installations/MDMversionJBOSSEAPSQL_COMMON/
   dynservices
   ```
6. Restart the server in case of cluster setup.

### What to do next

Log in to TIBCO MDM and check the pages.

## Deploying TIBCO MDM on JBoss WildFly Application Server

You need to deploy TIBCO MDM on JBoss WildFly Application Server.

### Procedure

1. Copy the `ECM.ear` file from `$MQ_HOME` to the `$JBOSS_HOME/standalone/deployments` directory.
2. To improve the performance of TIBCO MDM, add the following JVM parameter while starting the JBoss WildFly Application Server:

   ```
   -Dorg.apache.xml.dtm.DTMManager="org.apache.xml.dtm.ref.DTMManagerDefault"
   ```

   If the output does not contain any error or exception messages, you are ready to use the JBoss.

   When the log4j service of TIBCO MDM initializes, it overrides the JBoss `rootLogger` property and starts writing server logs in the `elink.log` file. As a workaround, you can comment out the `rootLogger` property in the `$MQ_HOME/config/ConfigValues.xml` file:

   ```
   <ConfValue description="The root logging level for the MDM server."
   name="Root logging Level for MDM Server" propname="log4j.rootLogger"
   sinceVersion="7.0" visibility="All"> <ConfString default="DEBUG"
   value="DEBUG"/> </ConfValue>
   ```

   It is recommended that you keep this property to change the root logging level for TIBCO MDM.

## Starting JBoss WildFly Application Server

After you configure TIBCO MDM with JBoss WildFly Application Server, you can start the JBoss WildFly Application Server.

### Procedure

1. On the command line, type *`$JBOSS_HOME`*/bin.
2. Enter the following command:

   - For local host: **`standalone.bat or ./standalone.sh`**
   - For remote server: **`standalone.bat -b 0.0.0.0 or ./standalone.sh -b 0.0.0.0`**

     To access the TIBCO MDM instance remotely, you need to provide **`-b 0.0.0.0`** in addition.

   The JBoss WildFly Application Server starts.

## Troubleshooting with JBoss WildFly Application Server

You may come across some exceptions on JBoss WildFly Application Server.

*Troubleshooting with JBoss Application Server*

| Issue | Description | Solution |
|---|---|---|
| Error occurs on the console for the unsupported Java versions | TIBCO MDM is installed with the Windows, JBoss Wildfly (16.0.0) application server, and Oracle database with the inbuilt Java version.<br>The JBoss WildFly application server started successfully, however, the error messages are displayed on the console for the unsupported Java versions. | TIBCO MDM supports JDK 11 version. Ensure that the `PATH` variable does not point to another Java version. |

# Configuring TIBCO MDM for WebSphere Application Server

Log in to the Administrative console of WebSphere Application Server to configure the properties.

Configure the following properties in the Configurator:

*Configurator Properties for WebSphere Application Server*

| Property Name | Value |
|---|---|
| JNDI Naming Service URL | By default, the value is `iiop://localhost:2809`. As per your application profile, you need to change the host IP address and port number. For the port number, refer to `BOOTSTRAP ADDRESS` specified in the WebSphere Application Server. Login to the Administrative console of the WebSphere Application Server console and expand **Application Servers** > ***server name*** > **Ports**. |
| Security Provider Type | By default, `SUN` security provider is defined. For WebSphere Application Server, you need to change it to `IBM`. |

For clustered setup, it is recommended that you:

- Increase the poolsize per server to `100`.
- Increase the transaction timeout of the application server to `36000`.

### Procedure

1. Creating a Profile on WebSphere Application Server
   a) Create a profile other than the default using the profile creation wizard.
   b) Start the Administration Server.
   - Go to the `$WAS_HOME/profiles/`*profilename*/bin directory.
   - Enter the following command: ./startServer.sh server1
2. Log in to the Administrative console of WebSphere Application Server.

3. Specify Servers Details
   a) In the left panel, expand **Servers** > **Server Types** and click **WebSphere** > **application servers** . The Application servers panel is displayed on the right.
   b) Under Preferences, click *servername*. The **Configuration** tab is displayed.
4. For Transaction Service Details:
   a) Under **Container Settings**, expand **Container Services** and click the **Transaction Service** link. The **Configuration** tab for the Transaction Service is displayed.
   b) Under General Properties, enter the following values:

   ***Transaction Service General Properties***

| Field / Drop-down List Name | Values |
|---|---|
| Total transaction lifetime timeout | Enter `36000`. |
| Client inactivity timeout | Enter `7200` |
| Maximum transaction timeout | Enter `0`. |
| Heuristic retry limit | Enter `0`. |
| Heuristic retry wait | Enter `0`. |
| Heuristic completion direction | Select **ROLLBACK**. |

   c) Click **OK**. A message is displayed with the Save and Review options.
   d) Click the **Save** link to save changes to the master configuration.
5. For ORB Service Details:
   a) Under Container Settings, expand **Container Services** and click the **ORB service** link. The **Configuration** tab is displayed.
   b) Under General Properties, check the **Pass by reference** check box.
   c) Click **OK**. A message is displayed with the Save and Review options.
   d) Click the **Save** link to save changes to the master configuration.
6. For Server Infrastructure Details:
   a) Under Server Infrastructure, expand **Java and Process Management** and click the **Process definition** link. The **Configuration** tab is displayed.
   b) Under Additional Properties, click the **Java Virtual Machine** link. The **Configuration** tab is displayed.
   c) In the Initial heap size and Maximum heap size fields, enter the heap size to **1024** as the minimum value for both the fields.
   d) In the Generic JVM arguments field, enter the following JVM arguments:
   e) For readability, each entry is listed on a separate line. However, you need to enter these arguments in a single line, separated by a single space.

```
-DLANG=en_US.UTF-8
-DNODE_ID=${NODE_ID}
-Dclient.encoding.override=UTF-8
-DMQ_HOME=${MQ_HOME}
-DMQ_LOG=${MQ_LOG}
-DMQ_CONFIG_FILE=${MQ_HOME}/config/ConfigValues.xml
-DMQ_COMMON_DIR=${MQ_COMMON_DIR}
-DPATH=${PATH}
-DOS=<OS>
-DDISPLAY=:1.0
-
Djavax.xml.transform.TransformerFactory=org.apache.xalan.processor.TransformerF
```

```
actoryImpl
-
Djavax.xml.parsers.DocumentBuilderFactory=org.apache.xerces.jaxp.DocumentBuilde
rFactoryImpl
```

- For the Oracle database, enter `-DORACLE_HOME=${ORACLE_HOME}`.
- To improve the TIBCO MDM performance, enter the following JVM parameter while starting the application server: `-Dorg.apache.xml.dtm.DTMManager="org.apache.xml.dtm.ref.DTMManagerDefault"`
- To deploy TIBCO MDM custom pages to a common folder in the cluster environment, set the following parameter: `-DMQ_CUSTOMFORMS_HOME=/home/apps/MDM_Installations/MDMversionWASORA_COMMON`

f) To monitor the JVM application server, enter the following JVM arguments:

```
-Dcom.sun.management.jmxremote
 -Dcom.sun.management.jmxremote.authenticate=false
-Dcom.sun.management.jmxremote.ssl=false
-Dcom.sun.management.jmxremote.port=9999
 -Djavax.management.builder.initial=
```

The value of `Djavax.management.builder.initial` argument must be empty.

g) Click **OK**. A message is displayed with the Save and Review options.

h) Click the **Save** link to save changes to the master configuration.

7. For Port Details: if multiple servers and clusters are involved while installing TIBCO MDM on the WebSphere Application Server, you need to configure and allocate multiple ports to application.

a) Under Communications, expand **Ports.** A list of ports is displayed.

b) Verify the value of the wc_defaulthost port. The WebSphere Application Server runs on this default port.

c) Ensure that the port is defined as a domain name system (DNS) alias in the *default_host* Virtual Host definition.

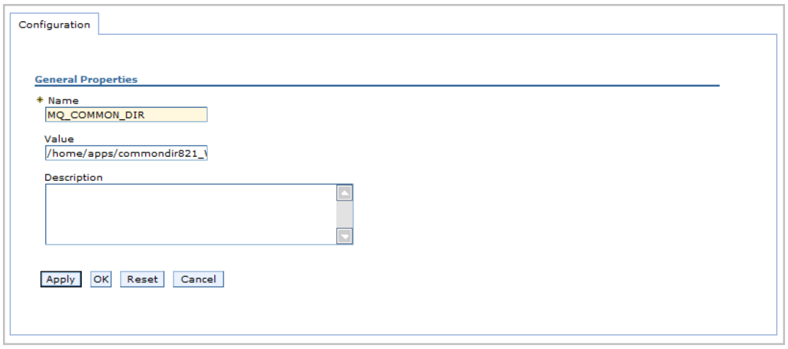
The DNS alias by which the virtual host is known is defined through **Environment** > **Virtual Hosts** > ***default_host*** > **Host Aliases**. The Virtual host for the Web modules that are contained in application is specified through **Applications** > **WebSphere Enterprise Applications** > ***ECM*** > **Virtual hosts**.

8. Specify Environment Variables

a) In the left panel, expand **Environment** and click **WebSphere variables** > **.** The WebSphere Variables panel is displayed on the right.

b) In the Scope drop-down list, select **Node=<nodeID>,Server=server1**.

c) Under Preferences, click **New**. The **Configuration** tab is displayed.



Configuration

General Properties

* Name
MQ_COMMON_DIR

Value
/home/apps/commondir821_\

Description

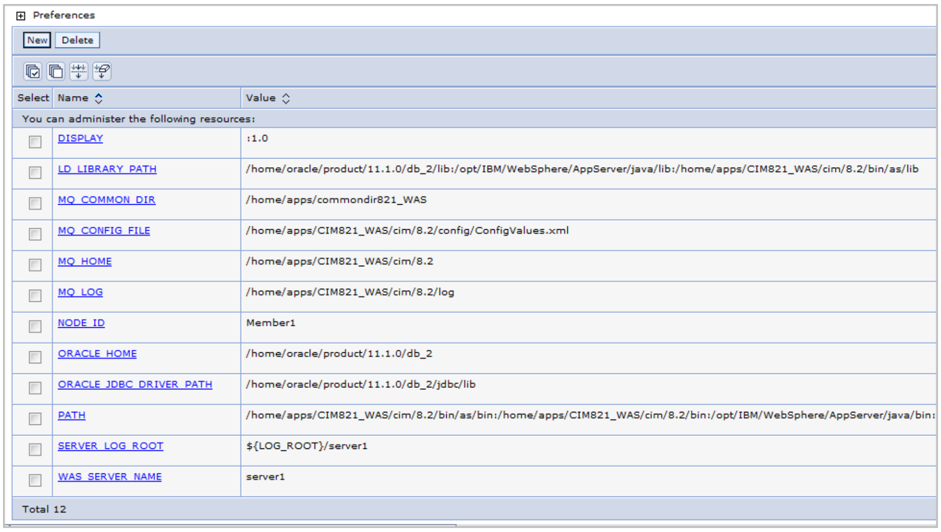
d) Under General Properties, type a system variable name in the Name field and its value in the Value field. The Description field is optional.

Add the following environment variable for the custom pages deployment:

```
MQ_CUSTOMFORMS_HOME=/home/apps/MDM_Installations/MDMversionWASORA_COMMON
```

e) Click **OK**. The variable is listed in the table.

Similarly, create other Environmental Variables. The following table displays a list of all other added environment variables.

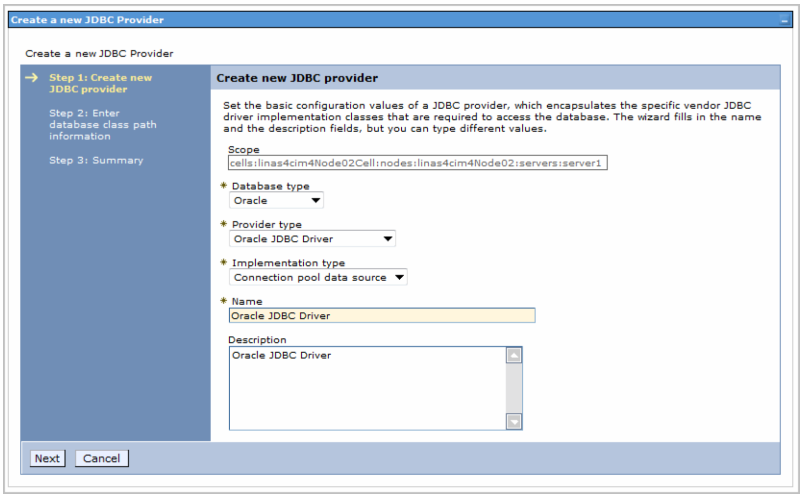


| Select | Name ⇕ | Value ⇕ |
|---|---|---|
| You can administer the following resources: | | |
| □ | DISPLAY | :1.0 |
| □ | LD_LIBRARY_PATH | /home/oracle/product/11.1.0/db_2/lib:/opt/IBM/WebSphere/AppServer/java/lib:/home/apps/CIM821_WAS/cim/8.2/bin/as/lib |
| □ | MQ_COMMON_DIR | /home/apps/commondir821_WAS |
| □ | MQ_CONFIG_FILE | /home/apps/CIM821_WAS/cim/8.2/config/ConfigValues.xml |
| □ | MQ_HOME | /home/apps/CIM821_WAS/cim/8.2 |
| □ | MQ_LOG | /home/apps/CIM821_WAS/cim/8.2/log |
| □ | NODE_ID | Member1 |
| □ | ORACLE_HOME | /home/oracle/product/11.1.0/db_2 |
| □ | ORACLE_JDBC_DRIVER_PATH | /home/oracle/product/11.1.0/db_2/jdbc/lib |
| □ | PATH | /home/apps/CIM821_WAS/cim/8.2/bin/as/bin:/home/apps/CIM821_WAS/cim/8.2/bin:/opt/IBM/WebSphere/AppServer/java/bin: |
| □ | SERVER_LOG_ROOT | ${LOG_ROOT}/server1 |
| □ | WAS_SERVER_NAME | server1 |
| Total 12 | | |

9. Configuring Database Drivers and Data Source

   a) Specify JDBC Details

      1. In the left panel, expand **Resources** > **JDBC** , and then click **JDBC providers**. The JDBC providers panel is displayed on the right.
      2. In the Scope drop-down list, select **Node=<nodeID>, Server=server1**.
      3. Under Preferences, click **New**. The Create a new JDBC Provider window is displayed.



Create a new JDBC Provider
Create a new JDBC Provider
→ Step 1: Create new JDBC provider
Step 2: Enter database class path information
Step 3: Summary
Next
Cancel

Create new JDBC provider

Set the basic configuration values of a JDBC provider, which encapsulates the specific vendor JDBC driver implementation classes that are required to access the database. The wizard fills in the name and the description fields, but you can type different values.

Scope: cells:linas4cim4Node02Cell:nodes:linas4cim4Node02:servers:server1

* Database type: Oracle

* Provider type: Oracle JDBC Driver

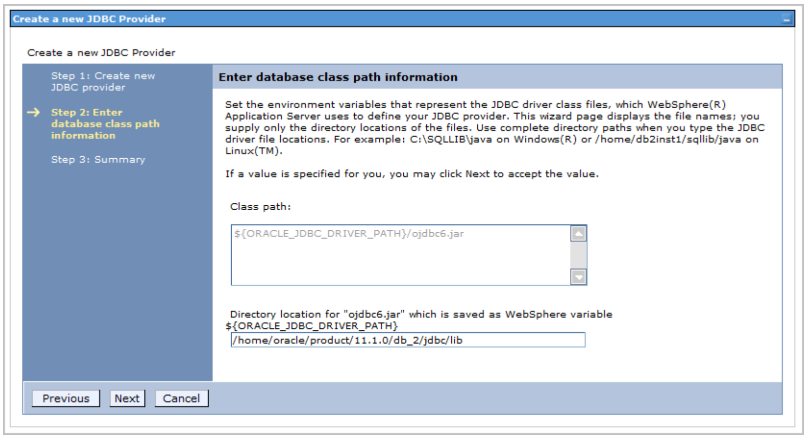
* Implementation type: Connection pool data source

* Name: Oracle JDBC Driver

Description: Oracle JDBC Driver

- In the Database type drop-down list, select **Oracle**. The Provider type is populated with **Oracle JDBC Driver.**
- In the Implementation type drop-down list, select **Connection pool data source**. The Name and Description fields are populated with Oracle JDBC Driver.

1. Click **Next**. The Enter database class path information window is displayed.



Create a new JDBC Provider

Create a new JDBC Provider

Step 1: Create new JDBC provider
→ Step 2: Enter database class path information
Step 3: Summary

**Enter database class path information**

Set the environment variables that represent the JDBC driver class files, which WebSphere(R) Application Server uses to define your JDBC provider. This wizard page displays the file names; you supply only the directory locations of the files. Use complete directory paths when you type the JDBC driver file locations. For example: C:\SQLLIB\java on Windows(R) or /home/db2inst1/sqllib/java on Linux(TM).

If a value is specified for you, you may click Next to accept the value.

Class path:

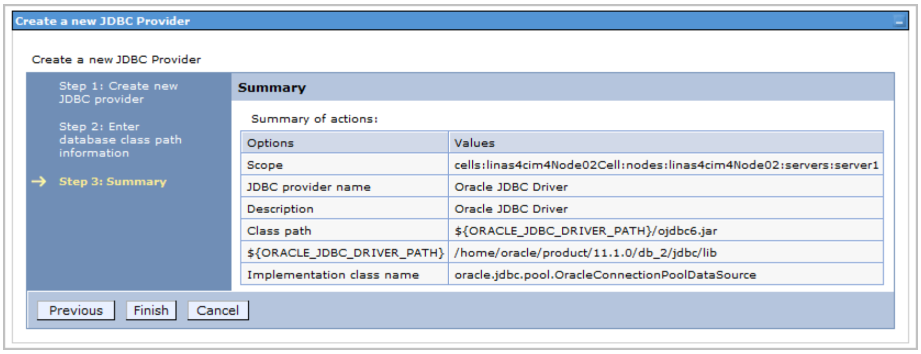
${ORACLE_JDBC_DRIVER_PATH}/ojdbc6.jar

Directory location for "ojdbc6.jar" which is saved as WebSphere variable ${ORACLE_JDBC_DRIVER_PATH}

/home/oracle/product/11.1.0/db_2/jdbc/lib

2. If you have selected the `Oracle database` option in the last step, type the location of the `ojdbc8.jar` file. If you are using Oracle Database 12.1, copy `ojdbce8.jar` in `$ORACLE_HOME/12c/lib.`

3. Click **Next**. The Summary window is displayed. Review the information.



Create a new JDBC Provider

Create a new JDBC Provider

Step 1: Create new JDBC provider
Step 2: Enter database class path information
→ Step 3: Summary

**Summary**

Summary of actions:

| Options | Values |
|---|---|
| Scope | cells:linas4cim4Node02Cell:nodes:linas4cim4Node02:servers:server1 |
| JDBC provider name | Oracle JDBC Driver |
| Description | Oracle JDBC Driver |
| Class path | ${ORACLE_JDBC_DRIVER_PATH}/ojdbc6.jar |
| ${ORACLE_JDBC_DRIVER_PATH} | /home/oracle/product/11.1.0/db_2/jdbc/lib |
| Implementation class name | oracle.jdbc.pool.OracleConnectionPoolDataSource |

4. Click **Finish**. The Oracle JDBC Driver is listed under Preferences and a message is displayed with the Save and Review options.

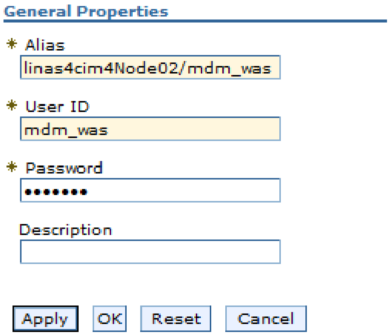
5. Click the **Save** link to save changes to the master configuration.

b) Specify Security Details

1. In the left panel, expand **Security** and click **Global security**. The Global security panel is displayed on the right.

2. Under Authentication, expand **Java Authentication and Authorization Service** and click the **J2C authentication data** link.

3. Under Preferences, click **New**. The General Properties window is displayed.



General Properties

* Alias
linas4cim4Node02/mdm_was

* User ID
mdm_was

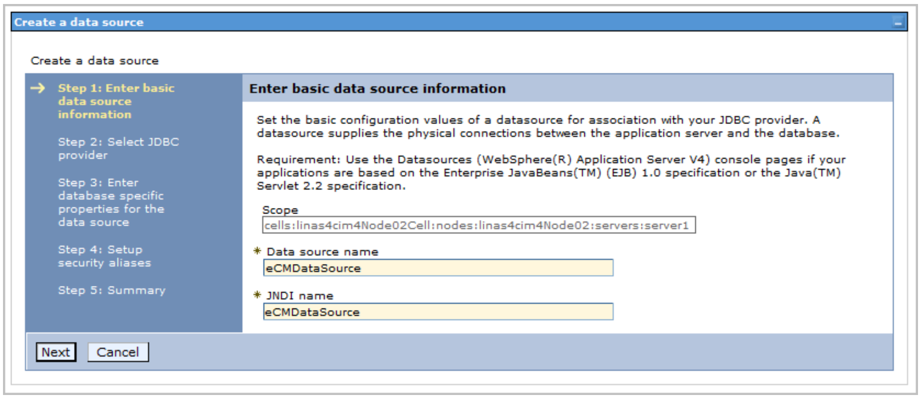
* Password
•••••••

Description

4. In the Alias, User ID, and Password fields, type the database alias name, its user ID, and password respectively. The Description field is optional.
5. Click **Apply**, and then click **OK**. The database Alias is listed under Preferences and a message is displayed with the Save and Review options.
6. Click the **Save** link to save changes to the master configuration.

c) Specify Data Source Details

1. In the left panel, expand **Resources** > **JDBC** , and then click **Data sources**. The JDBC providers panel is displayed on the right.
2. In the Scope drop-down list, select **Node=<nodeID>, Server=server1**.
3. Under Preferences, click **New**. The Create a data source window is displayed.



Create a data source

Create a data source

→ Step 1: Enter basic data source information
Step 2: Select JDBC provider
Step 3: Enter database specific properties for the data source
Step 4: Setup security aliases
Step 5: Summary

Enter basic data source information

Set the basic configuration values of a datasource for association with your JDBC provider. A datasource supplies the physical connections between the application server and the database.

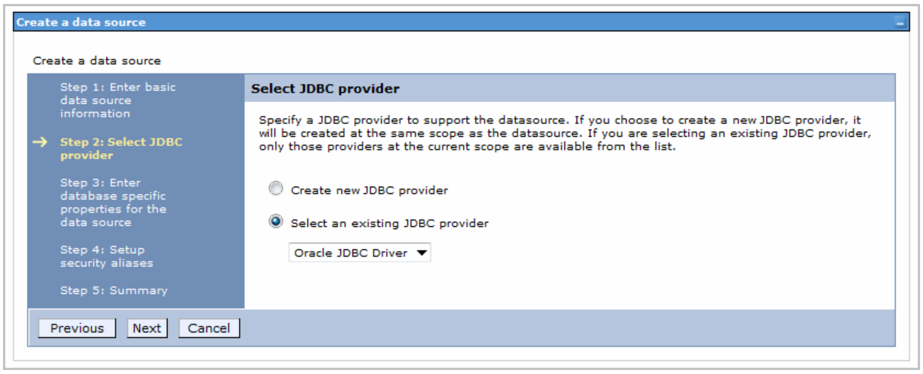
Requirement: Use the Datasources (WebSphere(R) Application Server V4) console pages if your applications are based on the Enterprise JavaBeans(TM) (EJB) 1.0 specification or the Java(TM) Servlet 2.2 specification.

Scope
cells:linas4cim4Node02Cell:nodes:linas4cim4Node02:servers:server1

* Data source name
eCMDataSource

* JNDI name
eCMDataSource

4. Enter the **Data source name** and **JNDI name**.
5. Click **Next**. The Select JDBC provider window is displayed.



Create a data source

Create a data source

Step 1: Enter basic data source information
→ Step 2: Select JDBC provider
Step 3: Enter database specific properties for the data source
Step 4: Setup security aliases
Step 5: Summary

Select JDBC provider

Specify a JDBC provider to support the datasource. If you choose to create a new JDBC provider, it will be created at the same scope as the datasource. If you are selecting an existing JDBC provider, only those providers at the current scope are available from the list.

○ Create new JDBC provider

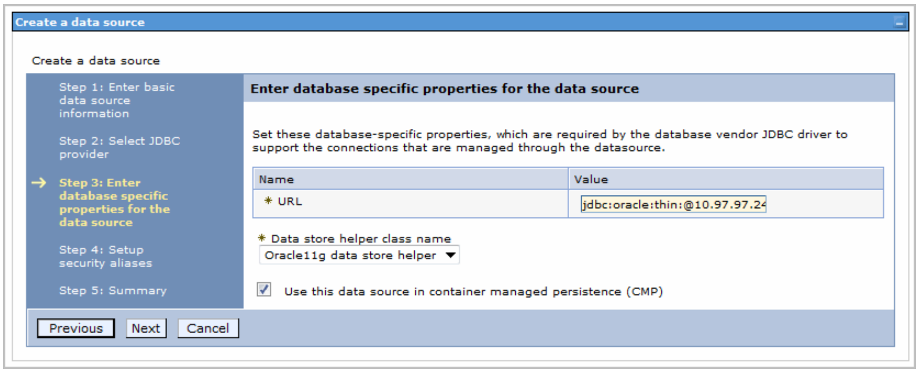
● Select an existing JDBC provider

Oracle JDBC Driver ▼

6. Select either of the following two options:

   - **Create a new JDBC provider**
   - **Select an existing JDBC provider**: After you select this option, the existing JDBC providers are displayed in the drop-down list. You can select it from the list.

7. Click **Next**. The Enter database specific properties for the data source window is displayed.


Create a data source
Create a data source
Step 1: Enter basic data source information
Step 2: Select JDBC provider
Step 3: Enter database specific properties for the data source
Step 4: Setup security aliases
Step 5: Summary
Previous
Next
Cancel

Enter database specific properties for the data source

Set these database-specific properties, which are required by the database vendor JDBC driver to support the connections that are managed through the datasource.

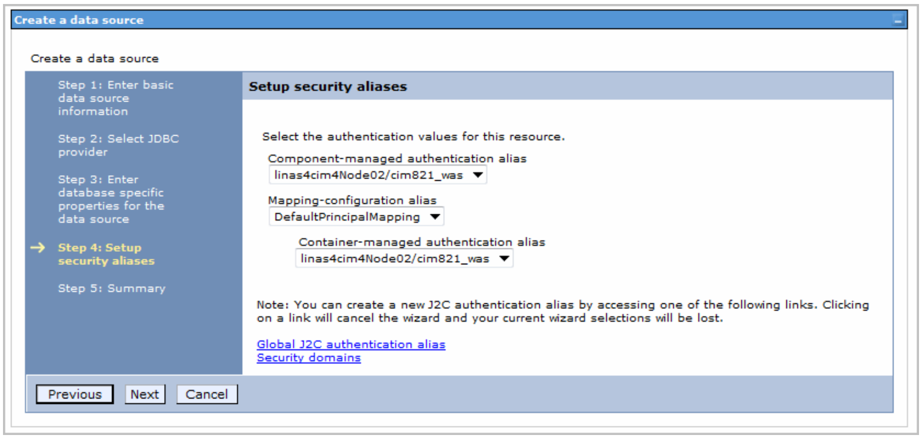
| Name | Value |
|---|---|
| * URL | jdbc:oracle:thin:@10.97.97.24 |

* Data store helper class name
Oracle11g data store helper

☑ Use this data source in container managed persistence (CMP)

   - In the Value field, enter the database connection URL. For example, for Oracle database: **jdbc:oracle:thin:@** *machinename* or **ipaddress**:*portnumber*: *INSTANCENAME*.
   - In the Data store helper class name drop-down list, select the appropriate data store helper class name. For example, **Oracle11g data store helper**.

8. Click **Next**. The Setup security aliases window is displayed.


Create a data source
Create a data source
Step 1: Enter basic data source information
Step 2: Select JDBC provider
Step 3: Enter database specific properties for the data source
Step 4: Setup security aliases
Step 5: Summary
Previous
Next
Cancel

Setup security aliases

Select the authentication values for this resource.

Component-managed authentication alias
linas4cim4Node02/cim821_was

Mapping-configuration alias
DefaultPrincipalMapping

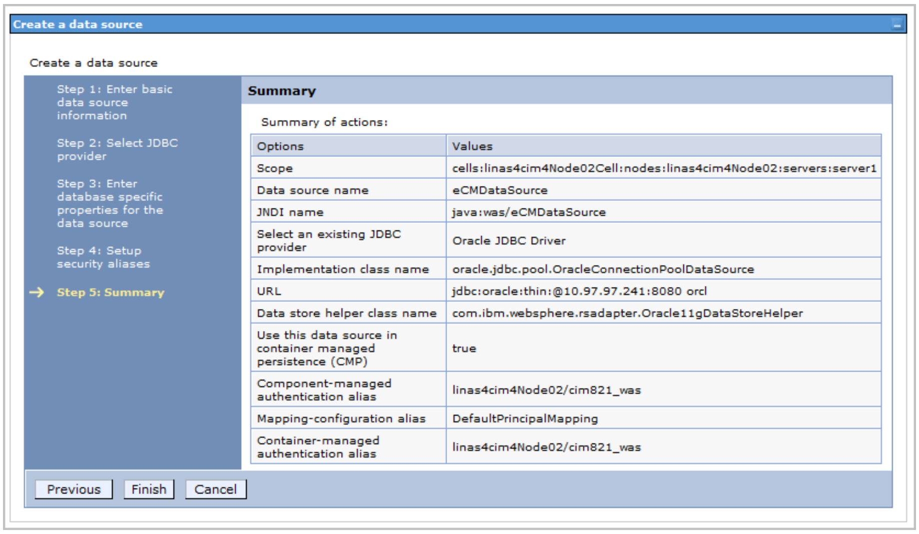
Container-managed authentication alias
linas4cim4Node02/cim821_was

Note: You can create a new J2C authentication alias by accessing one of the following links. Clicking on a link will cancel the wizard and your current wizard selections will be lost.

Global J2C authentication alias
Security domains

   - In the Component-managed authentication alias drop-down list, select the alias.
   - In the Mapping-configuration alias drop-down list, select **DefaultPrincipleMapping**.
   - In the Container-managed authentication alias drop-down list, select **alias**.

9. Click **Next**. The Summary window is displayed. Review the information.



Create a data source
Step 1: Enter basic data source information
Step 2: Select JDBC provider
Step 3: Enter database specific properties for the data source
Step 4: Setup security aliases
Step 5: Summary
Previous
Finish
Cancel

Create a data source

Summary

Summary of actions:

| Options | Values |
|---|---|
| Scope | cells:linas4cim4Node02Cell:nodes:linas4cim4Node02:servers:server1 |
| Data source name | eCMDataSource |
| JNDI name | java:was/eCMDataSource |
| Select an existing JDBC provider | Oracle JDBC Driver |
| Implementation class name | oracle.jdbc.pool.OracleConnectionPoolDataSource |
| URL | jdbc:oracle:thin:@10.97.97.241:8080 orcl |
| Data store helper class name | com.ibm.websphere.rsadapter.Oracle11gDataStoreHelper |
| Use this data source in container managed persistence (CMP) | true |
| Component-managed authentication alias | linas4cim4Node02/cim821_was |
| Mapping-configuration alias | DefaultPrincipalMapping |
| Container-managed authentication alias | linas4cim4Node02/cim821_was |

10. Click **Finish**. The data source is listed in the Preferences section and a message is displayed with the Save and Review options.

11. Click the **Save** link to save changes to the master configuration.

    > You must save the data source name before testing its connection, else an error message is displayed.

12. Under Preferences, select the *Data Source name* and click **Test Connection** to test the connection. A Connection Successful message is displayed.

    > Navigate to **Data sources** > *DataSourceName* > Connection pool properties to set the Maximum connections to 50 and the connection timeout to 7200.

## Deploying TIBCO MDM on WebSphere Application Server

After configuring TIBCO MDM with WebSphere Application Server, deploy TIBCO MDM.

### Procedure

1. In the left panel, expand **Applications** and click **New Application**. The New Application panel is displayed on the right.

2. Under Install a New Application, click the **New Enterprise Application** link. The Preparing for the application installation window is displayed.

3. Under Path to the new application, click **Browse**. The Choose File to Upload window is displayed.

4. Browse to the path of the ECM.ear file located in `$MQ_HOME`. Click **Next**.

    > If you have a ECM.ear file located at the remote location, you can select the file path using the Remote file system option.

5. Under How do you want to install the application?, select the **Detailed - Show all installation options and parameters** option, and then click **Next**. The Application Security Warnings are displayed.

6. Click **Continue**. The Install New Application window is displayed with the Select installation options dialog.

7. Click **Next**. The Map modules to server window is displayed. Verify that Cluster and servers are properly selected.

8. Click **Next**. The Provide JSP reloading options for Web modules window is displayed.

9. Click **Next**. The Map shared libraries window is displayed.
10. Click **Next**. The Map shared library relationships window is displayed.
11. Click **Next**. The Provide JNDI names for beans window is displayed. Verify that all JNDI names are pre-populated.
12. Click **Next**. The Map virtual hosts for Web modules window is displayed.
13. Click **Next**. The Map context roots for Web modules window is displayed.
14. Click **Next**. The Ensure all unprotected 2.x methods have the correct level of protection window is displayed with installation options summary.
15. Click **Finish**.
16. After installing, click **Save**.
    After a successful startup, a confirmation message is displayed in the log file located at `$WAS_HOME/logs`.

    > You can also verify if the application is installed successfully using the following URL: http://*hostname*:*port_number*/eml/Login, For example: http://localhost:9081/eml/Login

## Setting Class Loader Policy

After deploying TIBCO MDM on Websphere Application Server, set the class loader policy.

### Procedure

1. In the left panel, expand **Servers** > **Server Types** and click **WebSphere application servers**.
   The Application servers panel is displayed on the right.
2. Under Preferences, click *servername*.

   For example, **server1**. The **Configuration** tab is displayed.
3. Under Applications, click the **Installed applications** link.
4. Under Preferences, click the *application name*. For example, **ECM.**
5. Under Detail Properties, click the **Class loading and update detection** link.
   The **Configuration** tab is displayed.
6. Under Class Loader order, select the **Classes loaded with local class loader first (parent last)** option.
7. Under **WAR class loader policy**, ensure that the **Class loader for each WAR file in application** option is selected.
8. Click **Apply**, and then click **OK**.
   The class loader policy settings are saved a message is displayed with the Save and Review options.
9. Click the **Save** link to save changes to the master configuration.

## Specifying MIME Types

After Setting the class loader policy to paren last, configure the MIME types.

### Procedure

1. In the left panel, expand **Environment** and click **Virtual Hosts**. The Virtual Hosts panel is displayed on the right.
2. Under Preferences, click the virtual host link where ECM is installed. For example, **default_host**.
   The **Configuration** tab is displayed.

3. Under Additional Properties, click the **MIME Types** link. A list of MIME types is displayed.
4. Under Preferences, click the **New** button. The **Configuration** tab is displayed.
5. Under General Properties**,**

    - In the MIME Type field, enter **application/xml**.
    - In the Extensions field, enter **xslt**.

6. Click the **OK** button. A message is displayed with the Save and Review options.
7. Click the **Save** link to save changes to the master configuration.

## Setting Up Security

Specify Security related settings.

Perform the following tasks:

- Enabling Cookies
- Enabling URL Rewriting
- Setting Up SSL

## Enabling Cookies

TIBCO MDM uses a cookie to keep track of menus selected by the user. The business sensitive information stored in the cookies should be kept confidential and sent only over a secure link. Make cookies secure by requiring them to be transmitted only over secure links and to the appropriate location.

### Procedure

1. In the left panel, expand **Servers** > **Server Types** and click **WebSphere application servers**. The Application servers panel is displayed on the right.
2. Under Preferences, click *servername*. The **Configuration** tab is displayed.
3. Under Container Settings, expand **Web Container Settings** and click the **Web container** link. The **Configuration** tab is displayed.
4. Under Additional Properties, click the **Session management** link. The **Configuration** tab is displayed.
5. Under General Properties, click the **Enable cookies** check box. The **Configuration** tab for Cookies is displayed.
6. For the JSESSIONID cookie property, enter the domain and path in the Cookie domain and Cookie path fields for which session tracking cookie should be sent.
7. Click the **Restrict cookies to HTTPS sessions** check box to restrict session cookies to HTTPS sessions.
8. Click the **OK** button. A message is displayed with the Save and Review options.
9. Click the **Save** link to save changes to the master configuration.

## Enabling URL Rewriting

You need to enable URL rewriting to send web service request with JessionID and if the installation does not support cookies.

TIBCO MDM recommends that cookies are enabled.

### Procedure

1. In the left panel, expand **Servers** > **Server Types** and click **WebSphere application servers**. The Application servers panel is displayed on the right.
2. Under Preferences, click *servername*. The **Configuration** tab is displayed.
3. Under Container Settings, click the **Session management** link. The **Configuration** tab is displayed.
4. Under General Properties,

   - Select the **Enable cookies** and **Enable URL rewriting** check boxes.
5. Click the **OK** button. A message is displayed with the Save and Review options.
6. Click the **Save** link to save changes to the master configuration.

## Setting Up SSL for WebSphere Application Server

SSL needs to be set up to access the application through a browser over the HTTPS protocol.

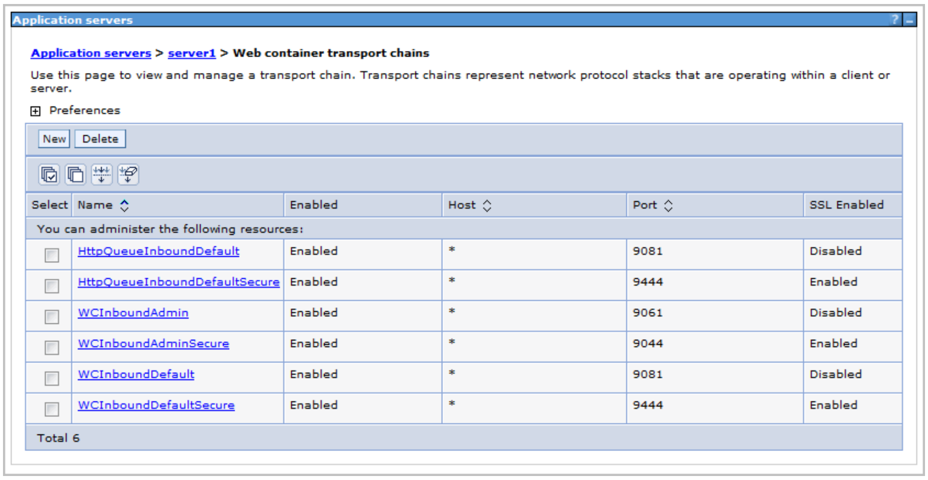
### Procedure

1. Enable Transport Chains

   a) In the left panel, expand **Servers** > **Server Types** and click **WebSphere application servers**. The Application servers panel is displayed on the right.

   b) Under Preferences, click *servername*. For example, **server1**. The **Configuration** tab is displayed.

   c) Under Container Settings, expand **Web Container Settings** and click the **Web container transport chains** link. The following screen displays the various ports on your server and also provides information on whether SSL is enabled. If SSL is not enabled, you can enable it.



Application servers > server1 > Web container transport chains

Use this page to view and manage a transport chain. Transport chains represent network protocol stacks that are operating within a client or server.

⊞ Preferences

| Select | Name | Enabled | Host | Port | SSL Enabled |
|---|---|---|---|---|---|
| You can administer the following resources: | | | | | |
| □ | HttpQueueInboundDefault | Enabled | * | 9081 | Disabled |
| □ | HttpQueueInboundDefaultSecure | Enabled | * | 9444 | Enabled |
| □ | WCInboundAdmin | Enabled | * | 9061 | Disabled |
| □ | WCInboundAdminSecure | Enabled | * | 9044 | Enabled |
| □ | WCInboundDefault | Enabled | * | 9081 | Disabled |
| □ | WCInboundDefaultSecure | Enabled | * | 9444 | Enabled |
| Total 6 | | | | | |

   d) Click **New**. The Create New Transport Chain window displays Select a transport chain template page.

   - In the Transport chain name field, enter transport chain name. For example, **MDMTransportChain2**.
   - In the Transport chain template drop-down list, select **WebContainer (templates/chains|webcontainer-chains.xml#Chain_1)**
     . For SSL, select **WebContainer-Secure(templates/chains|webcontainer-chains.xml#Chain_2)**.

   a) Click **Next**. The Create New Transport Chain window displays Select a port page.

- In the Port name field, enter the port name. For example, **MDMTransportChain2**.
- In the Host field, enter the host. For example, *.
- In the Port field, enter the port number. For example, **9082**. Use any unused port number.

a) Click **Next**. The Create New Transport Chain window displays Confirm new transport chain creation page.
b) Under Summary of Actions, a summary of the selections is displayed. Review the information.
c) Click **Finish**. The SSL Enabled column for the newly created TransportChain displays **Enabled** for the specified port and a message is displayed with the Save and Review options.
d) Click the **Save** link to save changes to the master configuration.

2. Specify Host Configuration

a) In the left panel, expand **Environment** and click Virtual Hosts. The Virtual Hosts panel is displayed on the right.
b) Under Preferences, click the virtual host link where ECM is installed. For example, **default_host**. The **Configuration** tab is displayed.
c) Under Additional Properties, click the **Host Aliases** link.
d) Under Preferences, click **New**.
e) Under General Properties**,**

- In the Host Name field, enter *.
- In the Port field, enter **9082** as port number or any other unused port number.

a) Click **OK**. A message is displayed with the Save and Review options.
b) Click the **Save** link to save changes to the master configuration.
c) Restart the server. Browse the application using the https protocol over the listening port.

3. Certificates (only applies if using GDSN software edition)

To communicate with 1SYNC securely, you may need to download their certificates and insert them into your trusted certificate store if they are not present already.

1. Open the web browser and type the secure URL of the server. For example, https://item.preprod.1worldsync.com.
2. If your computer does not have the certificates in its keystore, a security alert is displayed. This alert warns you that the certificate is not verified and allows you to view the certificate. Click **View Certificate** to view the certificate. The **Certificate** window is displayed.
3. Click on the **Certification Path** tab. Each certificate listed must be copied to a file.
   - Select the **Details** tab.
   - Click **Copy to File**. The **Certificate Export Wizard** opens and guides you through the steps for copying certificates, certificate trust lists, and certification revocation lists from a certification store to your disk.
   - Choose the **Base-64 encoded X.509 (.CER)** format and click **Next**.
   - Provide a filename and click **Next**.
   - Click **Finish**. The certificate is exported to the file.
4. Import the certificates into the trusted certificate store of the Java virtual machine. The default Java virtual machine is located at <Websphere install location>/AppServer/java, and the corresponding trusted certificate store is at <websphere install location>/AppServer/java/jre/lib/security/cacaerts.
5. Use the keytool utility in jre/bin/keytool to insert all the certificates. For example:

```
$keytool -import -alias 1sync -file ./1sync.cer -keystore ../lib/security/
cacerts -trustcacerts
```

The standard password for the cacerts store is changeit. After you confirm that you trust the certificate, the following message is displayed:

```
Certificate was added to keystore.
```

6. Repeat this procedure for all certificates.

For more information, see https://www.ibm.com/support/knowledgecenter/en/SSHS8R_6.3.0/com.ibm.worklight.installconfig.doc/appcenter/t_ac_ssl_was_full.html and https://www.ibm.com/support/knowledgecenter/prodconn_1.0.0/com.ibm.scenarios.wmqwassecure.doc/topics/cfgssl_was.htm .

## Troubleshooting with WebSphere Application Server

Resolve the errors that you may come across while working with the WebSphere Application Server.

*Troubleshooting with WebSphere Application Server*

| Issue | Description | Solution |
|---|---|---|
| Error Creating Catalogs and Data Sources After Installation | • Error during catalog creation through the UI.<br>• Catalog attributes not defined.<br>• Error during data source creation through the UI.<br>• Data source -1 could not be loaded. | Using the WebSphere Administrative Console, select **Servers** > **Application Servers** > **server1** > **Container Services** > **ORB Service** , then select the **Pass by Reference** check box.<br>**Background Information**: If these errors are seen while creating data sources and catalogs, the WebSphere configuration may be incorrect. In WebSphere, the ORB Service should have the **Pass by Reference** flag checked.<br>You can also check this value by viewing the WebSphere configuration `server.xml` file for your application server. Look at the following XML element and check if the **noLocalCopies** attribute is set to true:<br>`$WAS_HOME/profiles/<profile name>/config/cells/<cell name>/nodes/<node name>/servers/server1/server.xml`<br>`<services xmi:type="orb:ObjectRequestBroker" xmi:id="ObjectRequestBroker_<id>" enable="true" requestTimeout="180" requestRetriesCount="1" requestRetriesDelay="0" connectionCacheMaximum="240" connectionCacheMinimum="100" commTraceEnabled="false" locateRequestTimeout="180" forceTunnel="never" noLocalCopies="true">` |

| Issue | Description | Solution |
|---|---|---|
| Login Screen Not Visible and Logs Show “Naming service not available” Error | The Naming service not available error is displayed in the log file and the Login screen is not visible. | This usually happens when a wrong IIOP port number is specified in Configurator. When WebSphere starts, in the SystemOut.log, you should see the following output:<br>`[7/27/04 15:28:28:451 PDT]`<br>`7b04ccd1 HttpTransport A`<br>`SRVE0171I: Transport http is`<br>`listening on port 9,083.`<br>`[7/27/04 15:28:28:503 PDT]`<br>`7b04ccd1 RMIConnectorC A`<br>`ADMC0026I: RMI Connector`<br>`available at port 2810`<br>In this example, the `2810` port number should be used for the JNDI Naming Service URL property in Configurator (**Application Server** > **WEBSPHERE**). |
| Cannot Log In After Installation | You have the login screen and your seed data is good but you cannot log in. | This means your security provider is invalid. You can fix this by changing your security provider class name in the Configurator. The default security provider is SUN. If you are using WebSphere Application Server, select IBM as the security provider. |
| Enabling Memory Allocation Trace | Memory allocation tracing may be requested by TIBCO Customer Support for analysis of certain problems. | Set up the tracing as follows:<br>**Environment** > **WebSphere Variables.** Select the server.<br>Create the following environment entries:<br>`IBM_MALLOCTRACE - set value as 1`<br>`MALLOC_TRACE - set value to`<br>`$WAS_HOME/profiles/<profilename>/`<br>`logs/server1/mtrace.log`<br>Substitute the absolute directory name for `$WAS_HOME`. |
| Enabling Garbage Collection Data Logging | If you are experiencing memory usage issues, TIBCO Customer Support may request for collection of garbage collection statistics. | To enable garbage collection data collection, change the JVM settings as follows:<br>**Servers** > **Application Server** > <servername> > Server Infrastructure > Java and Process Management > Process Definition > Java Virtual Machine. Select the **Verbose garbage collection** check box against it.<br>The garbage collection data is stored in<br>`$WAS_HOME/profiles/<profilename>/`<br>`logs/server1/native_stderr.log` |

| Issue | Description | Solution |
|---|---|---|
| Failed Reflecting Values Error | The IWAV0002E Failed reflecting values warning is displayed when TIBCO MDM is installed on WebSphere Application Server. | Ignore this warning. For additional information, refer to the following site:<br>http://publib.boulder.ibm.com/infocenter/wchelp/v6r0m0/index.jsp?topic=/com.ibm.commerce.wcportal.doc/refs/rpo_configerror.htm |
| Incorrect Startup Message Error | When TIBCO MDM is deployed on the WebSphere Application Server, if for some reason the TIBCO MDM application does not start up because of initialization errors, the WebSphere Application Server UI still shows the application status as Started. | None. If the user in such a scenario hits the TIBCO MDM login page URL, initialization errors may be listed on that page. |
| The TIBCO MDM server failed to start | The TIBCO MDM server failed to start and showed the following error message: `java.lang.NoClassDefFoundError: javax.jms.JMSContext exception` | 1. Navigate to `$EMS_HOME/lib` directory.<br>2. Copy the `jms-2.0.jar` file.<br>3. Navigate to `$MQ_HOME/ECM.ear` and place the `jms-2.0.jar` file.<br>4. Navigate to `$MQ_HOME/ECM.ear/EML.war/META-INF` directory.<br>5. Open the `MANIFEST.MF` file and append the <space> `jms-2.0.jar` file name.<br>6. Save the `MANIFEST.MF` file.<br>7. Deploy the updated `ECM.ear` file in the WebSphere application server.<br>8. Restart the WebSphere application server. |
| An exception on the WebSphere Application Server startup | The `WebAppNotLoadedException` occurs on the WebSphere Application Server startup | After deploying TIBCO MDM on Websphere Application Server, ensure that the **Class loader for each WAR file in application** option is selected the class loader policy. For information, see the "Setting Class Loader Policy" section in *TIBCO MDM Installation and Configuration Guide*. |

## Configuring TIBCO MDM for WebLogic Application Server

Using the Configuration wizard, create a WebLogic domain.

**Prerequisites**

Configure the following properties in the Configurator:

*Configurator Properties for WebLogic Application Server*

| Property Name | Location | Value | Description |
|---|---|---|---|
| Application Server Name | *Node ID* > Application Server | WebLogic | The name of the application server. If the value is WebLogic, it refers to the WebLogic application server. |
| JNDI Naming Service URL | *Node ID* > Application Server > WebLogic | `t3://localhost:7001` | The Uniform Resource Locator (URL) used by the WebLogic application server to expose the J2EE Naming service. This value is has already been defined in the application server configuration. |
| Encryption Provider | InitialConfig > Security Provider > IBM | The default value is `com.ibm.crypto.provider.IBMJCE`. This value typically does not need to be changed. | Refers to the name of the Java class, which is the default security encryption on the IBM Java Virtual Machine. This value is sufficient. |
| JNDI Context Factory | *Node ID* > Application Server > WebLogic | The default value is `weblogic.jndi.WLInitialContextFactory`.This value typically does not need to be changed. | The Java class that initiates a connection to the naming service of the application server. |

### Procedure

1. Configure the Server
   a) Start the WebLogic application server.
   b) Verify that an instance of WebLogic is running by logging into the admin console (use the user name and password you provided during domain creation)
2. Set up Timeout Seconds
   a) Go to **Domain** > **Configuration** > **JTA** tab.
   b) Set the **Timeout Seconds** to **36000**.
   c) Click **Save**.
3. Enable the Archived Real Path to dynamically deploy the resource files of a customized page that is built using UI Builder and to improve the page loading performance.
   a) Go to **Domain** > **Configuration** > **Web Application** .
   b) Select the **Archived Real Path Enabled** check box.

c) Click **Save**.

4. Set up JDBC Drivers for Oracle - WebLogic

   a) In the left pane, expand **Services** > > **Data Sources** . The **Summary of JDBC Data Sources** screen is displayed.

   b) In the Data Sources section, click the **New** button to create a new data source. The following three options are displayed: **Generic Data Source**, **GridLink Data Source**, and **Multi Data Source**.

   c) Select the **Generic Data Source** option.

   d) In the **Name** field, specify a name for the JDBC Data Source.

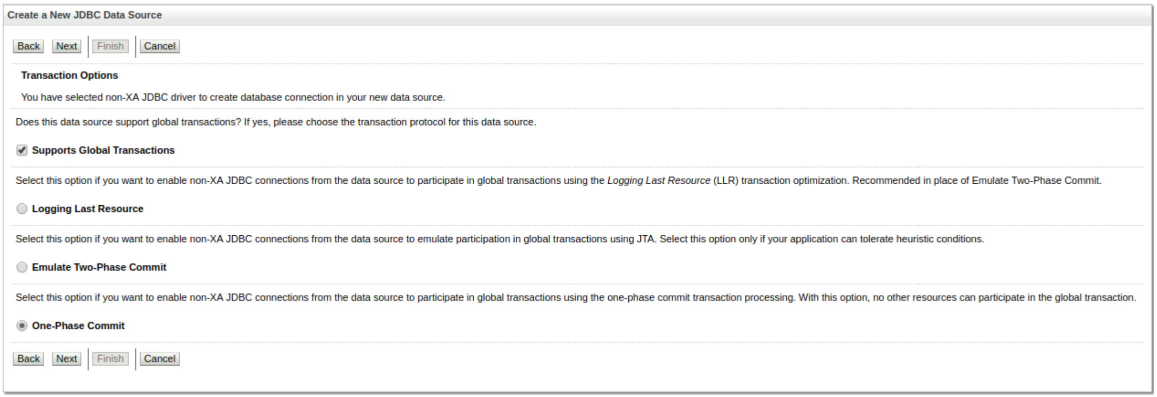
   > The name and JNDI name should be `eCMDataSource`.

   e) Select **Database Type** as **Oracle**.

   f) Click **Next**. The **Create a New JDBC Data Source** screen is displayed.

   g) Select **Oracle's Driver (Thin) for Service Connections; Versions: Any** from the Database Driver drop-down list.

   h) Click **Next**. Click **Next** on the following screen.



Create a New JDBC Data Source

**Transaction Options**

You have selected non-XA JDBC driver to create database connection in your new data source.

Does this data source support global transactions? If yes, please choose the transaction protocol for this data source.

☑ **Supports Global Transactions**

Select this option if you want to enable non-XA JDBC connections from the data source to participate in global transactions using the *Logging Last Resource* (LLR) transaction optimization. Recommended in place of Emulate Two-Phase Commit.

○ **Logging Last Resource**

Select this option if you want to enable non-XA JDBC connections from the data source to emulate participation in global transactions using JTA. Select this option only if your application can tolerate heuristic conditions.

○ **Emulate Two-Phase Commit**

Select this option if you want to enable non-XA JDBC connections from the data source to participate in global transactions using the one-phase commit transaction processing. With this option, no other resources can participate in the global transaction.

● **One-Phase Commit**

   i) Enter the details in the **Database Name, Host Name, Port, Database User Name**, **Password** and **Confirm Password** fields. Note that the `Database Name = SID`.

   j) Click **Next**.

   k) Click **Test Configuration**.

   l) Click **Next**. You are prompted to select targets to deploy your new JDBC data source.

   m) Select the Server and click **Finish**.

5. Configure the Connection Pool

   a) In the left pane, click **Services** > > **Data Sources** > > **eCMDataSource** . The **Settings for eCMDataSource** screen is displayed.

   b) Click the **Connection pool** tab.

   c) Expand **Advanced** at the bottom of the screen.

   d) Select the **Test Connections on Reserve** check box.

   e) Specify **7200** in the **Inactive Connection Timeout** field.

   f) Specify **10** in the **Maximum Waiting for Connection** field.

   > Maximum Capacity for connection settings is 100.

   g) Select the **Ignore In-Use Connections** and **Remove Infected Connections Enabled** check boxes.

   h) Click **Save**. The connection pool configuration for the eCMDataSource is saved.

   i) Copy the following files from `%TIBEMSDIR%\lib` to `%BEA_HOME%\user_projects\domains\`*`domain name`*`\lib` and `$MQ_HOME/lib/external`

   - 
     ```
     tibjms.jar
     ```

6. Modify the `startWebLogic.cmd` or `startWebLogic.sh` file.

a) Ensure that the WebLogic Application Server is running.

b) Edit the `startWebLogic.sh` or `startWebLogic.cmd` file located under `$BEA_HOME/user_projects/domains/`*domain_name*/bin to add the following lines in the `# START WEBLOGIC` section:

> If you are using Windows operating system, replace `$` with `%text%` in the following example:

```
java ${JAVA_VM} ${MEM_ARGS} ${JAVA_OPTIONS}
-Dweblogic.Name=${SERVER_NAME}
-Dweblogic.management.username=${WLS_USER}
-Dweblogic.management.password=${WLS_PW}
-Dweblogic.ProductionModeEnabled=${STARTMODE}
-DMQ_HOME=${MQ_HOME}
-DMQ_CONFIG_FILE="${MQ_HOME}/config/ConfigValues.xml"
-DMQ_COMMON_DIR=${MQ_COMMON_DIR}
-DORACLE_HOME=${ORACLE_HOME}
-DNODE_ID=Member1
-DMQ_LOG=${MQ_LOG}
-DTNS_ADMIN=${ORACLE_HOME}/network/admin
-DPATH=${MQ_HOME}/bin:${WL_HOME}/server/bin:${WL_HOME}/common/bin:$
{ORACLE_HOME}/bin:/usr/bin
-Djava.security.policy="${WL_HOME}/server/lib/weblogic.policy" weblogic.Server
```

> - To deploy the Configurator in WebLogic, set the following parameter: -`Dcatalina.home=${MQ_LOG}`
> - To deploy the TIBCO MDM custom pages to a common folder in the cluster environment, set the following parameter: `set MQ_CUSTOMFORMS_HOME=/home/apps/MDM_Installations/MDM`*`version`*`WLGORA_COMMON`
> - Add the following environment variable in the WebLogic startup section: `-DMQ_CUSTOMFORMS_HOME=/home/apps/MDM_Installations/MDM`*`version`*`WLGORA_COMMON`

a) Optional: To improve the TIBCO MDM performance, add the following JVM parameter while starting the application server:

```
-Dorg.apache.xml.dtm.DTMManager="org.apache.xml.dtm.ref.DTMManagerDefault"
```

b) To monitor application server JVM, add the following JVM arguments:

```
-Dcom.sun.management.jmxremote.authenticate=false
-Dcom.sun.management.jmxremote.ssl=false
-Dcom.sun.management.jmxremote.port=9999
```

c) Set up `TIBEMSDIR=`*path* environment variable for EMS installation. For example: set **`TIBEMSDIR=C:\Tibco\ems5.1\ems\5.1`**

d) The CLASSPATH set in the WebLogic startup script must include the location of the JAR files. Include the following JAR files in the *CLASSPATH* variable.

- Windows:

```
%TIBEMSDIR%\lib\tibjms.jar;
%TIBEMSDIR%\lib\tibjmsapps.jar;
%TIBEMSDIR%\lib\tibrvjms.jar;
%TIBEMSDIR%\lib\tibjmsadmin.jar;
%MQ_HOME%\lib\external\xbean.jar;
%MQ_HOME%\lib\external\log4j-1.2.17.jar;
%MQ_HOME%\lib\external\commons-logging-1.2.jar;
%MQ_HOME%\lib\external\gwt-user.jar
%MQ_HOME%\lib\external\dom4j-1.6.1.jar;
%MQ_HOME%\lib\external\hibernate\hibernate3.jar;
%MQ_HOME%\lib\external\javassist-3.12.0.GA.jar;
%MQ_HOME%\lib\external\slf4j-api-1.7.16.jar;
%MQ_HOME%\lib\external\stickyConfiguration.jar
```

- UNIX:

```
$TIBEMSDIR/lib/tibjms.jar:
$TIBEMSDIR/lib/tibjmsapps.jar:
$TIBEMSDIR/lib/tibrvjms.jar:
```

```
$TIBEMSDIR/lib/tibjmsadmin.jar:
$MQ_HOME/lib/external/xbean.jar:
$MQ_HOME/lib/external/log4j-1.2.17.jar:
$MQ_HOME/lib/external/commons-logging-1.2.jar:
$MQ_HOME/lib/external/gwt-user.jar
$MQ_HOME/lib/external/dom4j-1.6.1.jar:
$MQ_HOME/lib/external/hibernate/hibernate3.jar:
$MQ_HOME/lib/external/javassist-3.12.0.GA.jar:
$MQ_HOME/lib/external/slf4j-api-1.7.16.jar:
$MQ_HOME/lib/external/stickyConfiguration.jar:$CLASSPATH
```

## Deploying TIBCO MDM on WebLogic Application Server

After configuring TIBCO MDM with WebLogic Application Server, deploy TIBCO MDM.

### Procedure

1. Type the following command:

   **`$ ./ startWebLogic.sh or startWebLogic.cmd located under $BEA_HOME/user_projects/domains/domain_name/bin`**
2. Launch the WebLogic Server Console (for example: http://localhost:7001/console)
3. Log into the Console.
4. In the left pane, click **Deployments**.
5. Click **Install** under **Deployments** on the **Control** tab.
6. Browse to the location of the ECM.ear file.
7. Select `ECM.ear` and click **Next**.
8. Choose targeting style as **Install this deployment as an application**.
9. Click **Next**.
10. Click **Finish**.
    A success message is displayed.

- Ensure that you click **Activate Changes** on the left to activate all your changes (not applicable in case of WebLogic 10.3). Under Deployments, select the application and click **Start** and select **Servicing all requests** option.
- You can also verify if the application is installed successfully using the following URL: http://*IP address*:7001/eml/Login. The port for the WebLogic Application Server is 7001.

## Setting Up SSL for WebLogic Application Server

SSL needs to be setup to access the application through a browser over the HTTPS protocol.

### Procedure

1. Log into the WebLogic Server Console.
2. Go to **Environment** > **Servers** > *server name* from the navigation tree in the left pane.
3. Select the **General** tab.
4. Select the **SSL Listen Port Enabled** check box**.**
5. Specify the port number (**SSL Listen Port**) where the SSL port will be listening. The default port number is 7002.
6. Clear the **Listen Port Enabled** check box (to disable the HTTP protocol).
7. Select the **Keystores** and **SSL** tabs and ensure that you are using the default settings.

8. Logout of the console and restart the server.

### Result

You also need to change the JNDI Provider URL to Https://localhost:*port* using the Configurator.

For more information, see https://www.oracle.com/webfolder/technetwork/tutorials/obe/fmw/wls/12c/12_2_1/01-38-001-ConfiguringSSL/ConfiguringSSL.html.

## Troubleshooting with WebLogic Application Server

Resolve the errors that you may come across while working with the WebLogic Application Server.

*Troubleshooting with WebLogic Application Server*

| Issue | Description | Solution |
|---|---|---|
| Login Page is Not Shown After Installation | The Login Page is not shown and error.log shows an error. | Check for JDBC driver. It should be oracle.jdbc.driver.OracleDriver.<br>Specify `$AS_HOME/bin` and `$AS_HOME/lib` in the classpath. |
| 404 Page Not Found | You get the 404 Page Not Found error when using the Apache 2.0 Plug-in for the WebLogic Sever. | Check PathTrim property within `weblogic.conf`. It has to be null, otherwise it will trim the /eml part from the URL.<br>Check the `httpd.conf` file for the <IfModule mod_weblogic.c> section. The path given within the include statement for `weblogic.conf` is relative to the Apache20 directory. |
| Garbage Collection Data is Not Available for Analysis | If you are experiencing performance issues, TIBCO Customer Support may request for collection of garbage collections statistics. | To enable garbage collection data collection, change the JVM settings as follows:<br>**Servers > Application Server > server1 > Process Definition > Java Virtual Machine.** Select the **Verbose garbage collection** check box against it.<br>The garbage collection data is stored in `$WAS_HOME/profiles/<profilename>/logs/server1/native_stderr.log` |

# Configuration of Web Servers

You can choose to use Microsoft IIS, Apache, or IBM HTTP web server.

For instructions on how to setup other web servers, refer to the documentation provided by web server vendors.

## Configuring IBM HTTP Web Server with WebSphere

Install the web server, if not already present. You can configure IBM HTTP server for WebSphere Application Server.

For WebSphere Application Server, the plug-in file is at `$WAS_HOME/config/cells/plugin-cfg.xml`.

### Procedure

1. Copy the plugin file (plugin-cfg.xml) to any location on the computer where the webserver is installed and specify its path in the `httpd.conf` file with an entry:
   `<WebSpherePluginConfig /path/plugin-cfg.xml`.
   You need read write permissions to modify this configuration file.
2. Configure access for the web server to the plugin library specified by LoadModule ibm_app_server_http_module.

   For details on configuring web servers, see the WebSphere Information Center online documentation.

   > If for security reasons, you need to prevent server information (such as, Web Server and Application Server versions) from being transmitted in the header file, perform the following steps:
   >
   > 1. Edit the `httpd.conf` file in the Web Server conf directory. For example,
   >
   >    In the `/opt/IBMHttpServer/conf/httpd.conf` file add the following line:
   >
   >    ```
   >    ServerTokens Prod
   >    ```
   > 2. Restart the web server and application server.

## Configuring Apache Web Server Plug-in with WebLogic

You can configure Apache plug-in configuration for WebLogic Application Server.

### Procedure

1. Copy the mod_wl_*version*.so file to the *$APACHE_HOME* /modules folder.
2. Modify the `httpd.conf` file located in the *$APACHE_HOME*/conf directory. You need read write permission to modify this configuration file.
   a) Search for Dynamic Shared Object (DSO) Support within `Httpd.conf` and append the listing with:

   ```
   LoadModule weblogic_module modules/mod_wl_version.so
   ```
   b) Search for Bring in additional module-specific configurations within `Httpd.conf` and add the following lines:

   ```
   <IfModule mod_weblogic.c>
      Include conf/weblogic.conf
   </IfModule>
   ```
   c) Search for **server-info** within `httpd.conf` and add:

   ```
   <Location /eml>
     SetHandler weblogic-handler</Location>
   ```

3. Create a `weblogic.conf` file in the *$APACHE_HOME*/conf directory.
4. Add the following properties to the `weblogic.conf` file. Maintain a single space between a property name and property value:

   WebLogicHost *weblogic-host-name*

   WebLogicPort *weblogic-port*

   PathTrim null

## Testing Apache Web Server Plug-in

After you configure Apache plug-in for WebLogic Application Server, you can test for verification purpose.

### Procedure

1. Start the WebLogic application server.
2. Start the Apache web server.
3. Enter the URL as **http://***apache.server.com***/eml/Home**

   The login page is displayed.

# TIBCO MDM Containerization

You can containerize TIBCO MDM and run it in a Docker or Kubernetes environment. To containerize TIBCO MDM, you must build and run the Docker images using the bundled Docker ZIP file.

The Dockerfiles are delivered as a ZIP file on the TIBCO eDelivery website. Download the `TIB_mdm_9.2.0_container.zip` file and extract its content to a separate directory. In the directory, locate the ready-to-use Dockerfile and other scripts required to build the images. You can build images using the Dockerfiles, and then run them as containers. For information about building images, see Building and Running TIBCO MDM All-in-One Container Docker Image and Building TIBCO MDM Containers for Cluster Docker Images.

A container consists of an entire runtime environment: an application; all its dependencies, and configuration files needed to run it bundled into one package. You need not worry about the differences of operating system distribution in case of container application. For information about Docker concepts, such as, Dockerfile, Docker Image, and Container, see Docker documentation.

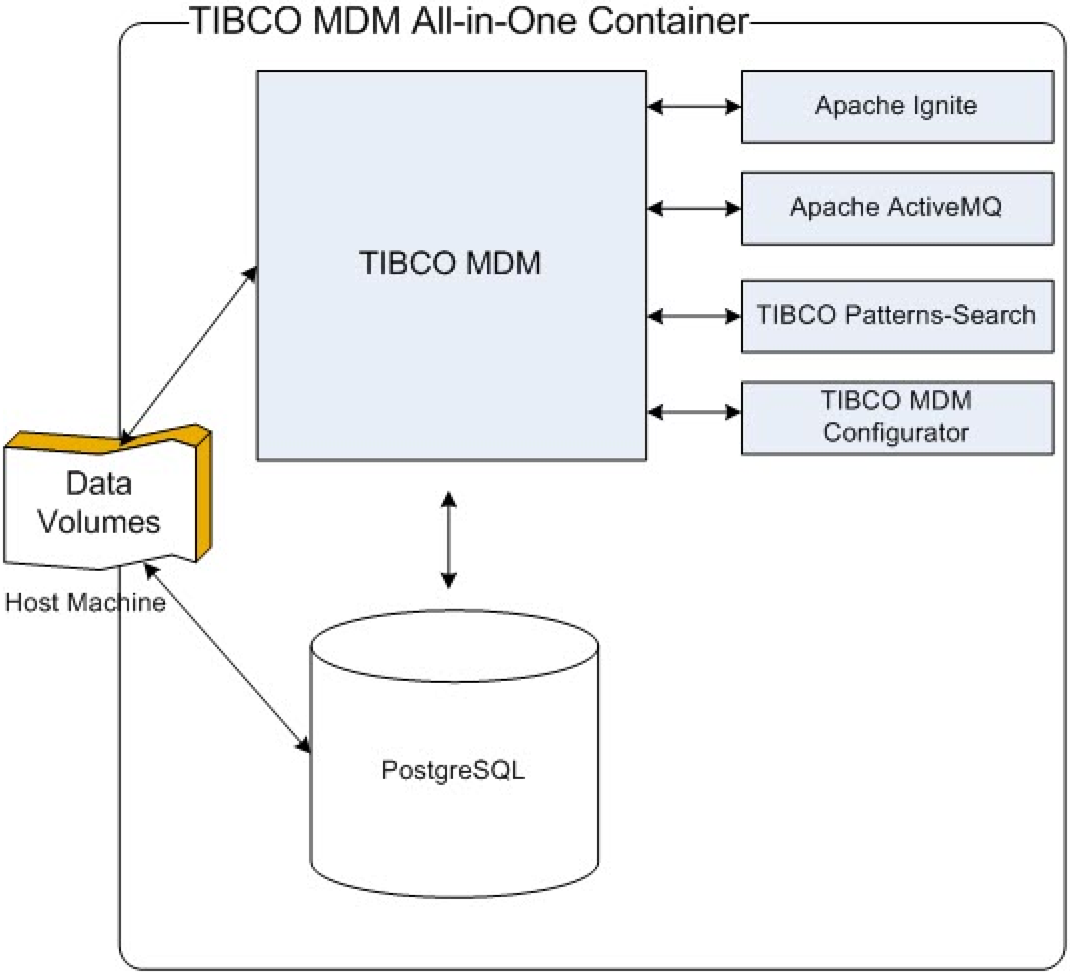
The TIBCO MDM containers are available in the following modes:

- Container for TIBCO MDM (All-in-One)
- Container for TIBCO MDM (Cluster)

Before you build and run the Docker image of TIBCO MDM All-in-One container, TIBCO MDM Cluster container, and the components included in the TIBCO MDM Cluster container, install Docker on the machine and perform the initial setup based on your operating system. For complete details on Docker installation, see Docker documentation.

## TIBCO MDM All-in-One Container

TIBCO MDM All-in-One container bundles components ( TIBCO MDM 9.2.0, TIBCO MDM Configurator, JBoss Wildfly 16.0.0, PostgreSQL 10.5, Apache ActiveMQ 5.15.7, Apache Ignite 2.7.0, and TIBCO Patterns - Search 5.4.0) as a single container. You can run the All-in-One container quickly by using only Docker, without complex configurations. You can use the TIBCO MDM All-in-One container in developement and QA enviroments for testing and demos.



The container is configured with seed data, which is ready to be used. TIBCO MDM All-in-One Container needs volumes to persist data. Docker volumes persist even if the container itself is stopped or deleted. You can re-initialize the container by dropping and recreating the volumes, without building the image again. TIBCO MDM All-in-One container supports only the PostgreSQL database.

## Building and Running the TIBCO MDM All-in-One Container Docker Image

Before you run TIBCO MDM All-in-One container, you must build a Docker image of it.

### Prerequisites

- Ensure that you have Dockerfile for TIBCO MDM All-in-One Container. Dockerfile is available in the `docker\AllInOne` folder.
- Enable `squash` as an **Experimental** feature through Docker configuration. For more information, see Docker documentation.

### Procedure

1. Copy the TIBCO MDM installer file (`TIB_mdm-JBOSS-PGRS_versionnumber_linux_x86_64.zip`) to the folder where the Dockerfile is located.
2. On the command line, enter the following command:

```
$> docker build -t mdm:versionnumber.GA --squash --rm=true.
```

3. Create the required Docker volumes by using the following commands:

   ```
   docker volume create --name mdmcommon
   docker volume create --name mdmconfig
   docker volume create --name mdmdynservices
   docker volume create --name postgresdata
   docker volume create --name mdmdbdata
   ```

4. On the command line, enter the following command to run the Docker container:

   ```
   --memory=container memory
   - Xms=jvm_minimum_memory
   - Xmx=jvm_maximum_memory
   ```

   For example:

   ```
   --memory=4096m
   - Xms=512m
   - Xmx=2048m
   ```

   ```
   docker run --memory=4096m
   -e JAVA_OPTS='-server -Xms512m -Xmx2048m -XX:MetaspaceSize=96M -
   XX:MaxMetaspaceSize=512m
   -Djava.net.preferIPv4Stack=true
   -Djboss.modules.system.pkgs=org.jboss.byteman
   -Djava.awt.headless=true
   -Djboss.as.management.blocking.timeout=3600'
   -p 8080:8080
   -p 6080:6080
   -v mdmcommon:/home/mdmuser/tibco/mdm/versionnumber/common
   -v mdmconfig:/home/mdmuser/tibco/mdm/versionnumber/config
   -v mdmdynservices:/home/mdmuser/tibco/mdm/versionnumber/dynservices
   -v postgresdata:/home/mdmuser/tibco/mdm/versionnumber/bin/pgsql/data
   -v mdmdbdata:/home/mdmuser/tibco/mdm/versionnumber/bin/pgsql/tablespaces
   mdm:versionnumber.GA
   ```

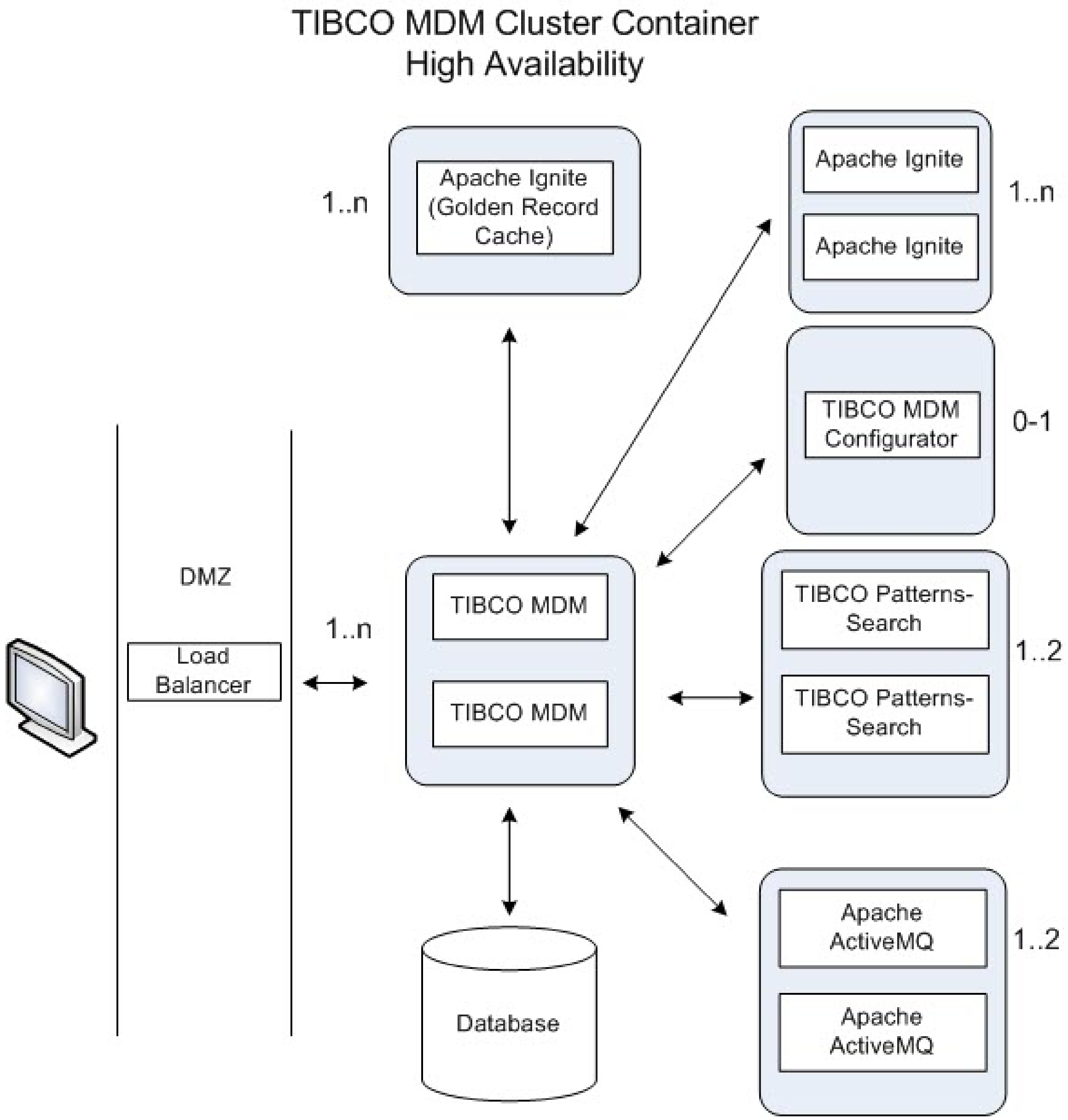
   You can specify minimum and maximum memory required by using `JAVA_OPTS`.

#### What to do next

Access the TIBCO MDM UI using the URL: http://*hostname*:8080/eml/Login

## TIBCO MDM Cluster

TIBCO MDM Cluster consists of the following containers: TIBCO MDM 9.2.0, TIBCO MDM Configurator 9.2.0, Apache ActiveMQ 5.15.7, Apache Ignite 2.7.0, and TIBCO Patterns - Search 5.4.0. You can scale up or down TIBCO MDM Server and Apache Ignite cache without user request being intercepted. Therefore, TIBCO MDM cluster container can be configured for high availability environment. You can use the TIBCO MDM Cluster in production environment for complex testing and demos.



The TIBCO MDM cluster supports PostgreSQL, Oracle, and Microsoft SQL Server databases. TIBCO Patterns-Search and Apache ActiveMQ servers can also be scaled as primary and secondary backup servers.

Kubernetes is required to run the TIBCO MDM cluster. Kubernetes is an orchestration engine for managing containerized applications across multiple hosts providing basic mechanisms for deployment, maintenance, and scaling of applications. For more information about Kubernetes, see Kubernetes Documentation.

## Building TIBCO MDM Cluster Container Docker Image

Before you run the TIBCO MDM cluster container, you must create a Docker image of it.

### Prerequisites

- Ensure that you have Dockerfile for the TIBCO MDM container. Dockerfile is available in the `docker\tropos\Mdm` folder.

- Ensure that you have the `mdmbase:latest` image built. To build the `mdmbase:latest` Docker image, see the `ReadMe.txt` file located at `docker\tropos\MdmBase`.

### Procedure

1. Copy the TIBCO MDM installer `TIB_mdm-JBOSS-PGRS_versionnumber_linux_x86_64.zip` file into the folder where the Dockerfile is located.
2. For the Oracle database, copy the database driver to the `jars` folder.
   a) Copy the driver JAR file in the `docker\tropos\Mdm\JBModules\modules\system\layers\base\com\oracle\jdbc\driver\main` and rename it to `ojdbc.jar`.
   b) Open the `module.xml` file from the `docker\tropos\Mdm\JBModules\modules\system\layers\base\com\oracle\jdbc\driver\main` and uncomment the `<resource-root>` element.
3. On the command line, enter the following command to build the Docker image:

```
docker build -t mdmc/mdm:versionnumber.latest --squash --rm=true
```

### What to do next

| | |
|---|---|
| Create TIBCO MDM database schema before using the application. | 1. On the command line, enter the following command to run the container:<br>`docker run -it mdmc/mdm:versionnumber.latest bash`<br>2. Copy the database seed data scripts from the Docker container to the host machine.<br>`docker cp containerID:/home/mdmuser/tibco/mdm/versionnumber/db /{host}/{path}/{target}`<br>To create seed data, you can use the files that you have copied from the Docker container based on the database that you have used for TIBCO MDM. For information about creating seed data, see the following sections:<br>• For Oracle: Creating Seed Data Manually for Oracle Database<br>• For PostgreSQL: Creating Seed Data Manually for PostgreSQL Database |
| Copy TIBCO MDM templates (customer, insurance, and party models) from the Docker container to the host machine | On the command line, enter the following command<br>`docker cp mdmContainerId:/home/mdmuser/tibco/mdm/version/templates/TemplateName/{host}/{path}/{target}`<br>OR<br>From the master node, run the following command:<br>`kubectl cp some-namespace/mdm-pod:/home/mdmuser/tibco/mdm/version/templates/{host}/{path}/{target}` |

## Building TIBCO MDM Containers for Cluster Docker Images

Create the Docker images for the components included in the TIBCO MDM Cluster.

### Prerequisites

- Create TIBCO MDM database schema.
- Ensure that you have a Dockerfile for each component for which you are creating the Docker image.

The steps to build TIBCO MDM Containers for Cluster Docker images are documented in the `ReadMe.txt` file available in each of the component folder.

Refer to the following table for the Dockerfile and readme location for each component:

| Docker Image (Component) Name | Dockerfile and Readme Location |
|---|---|
| TIBCO MDM Base | `docker\tropos\MdmBase` |
| ❗ You must first build the `mdmbase:latest` Docker image before you create the Apache Ignite, TIBCO Patterns - Search, Apache ActiveMQ , and TIBCO MDM Configurator Docker images. | |
| Apache Ignite | `docker\tropos\Ignite` |
| TIBCO Patterns-Search | `docker\tropos\Patterns` |
| Apache ActiveMQ | `docker\tropos\ActiveMQ` |
| TIBCO MDM Configurator | `docker\tropos\MdmConfig` |

## TIBCO MDM Cluster Container Components YAML Files

All YAML files are located in the `docker/k8s_deployment` folder. The YAML files define the Kubernetes objects that are required for deployment. You can update YAML files and deploy objects to the cluster to change configuration. Use YAML files to configure Kubernetes resources such as pods, services, and deployments.

| YAML File Name | Description |
|---|---|
| ConfigMap (`config_dep.yaml`) | A Config map stores configuration data for containers. Config map separates out configurations from your Pods and components. It is easier to change and manage config maps, without hardcoding configuration data to Pod specifications.<br>For the parameters available in the ConfigMap YAML file, see ConfigMap YAML File Parameters. |
| Secrets (`secrets_dep.yaml`) | Secrets are objects, which stores sensitive information about your clusters such as user name and password in the encrypted format. |

| YAML File Name | Description |
|---|---|
| Deployments | Deployment object consists of specification for Pods and defines ReplicaSets. If one of the instances of your application fails, it is replaced by another replica without user request being intercepted.<br>YAML deployment files for the following components are also available in the `docker\k8s_deployment` folder.<br>• TIBCO MDM: `mdm_dep.yaml`<br>• TIBCO MDM Configurator: `mdmconfig_dep.yaml`<br>• Apache Ignite: `ignite_dep.yaml`<br>• TIBCO Patterns - Search: `patterns_dep.yaml`<br>• Apache ActiveMQ: `activemq_dep.yaml`<br>In the deployment files, you can update values of replicas, image name, and memory.<br>⚠ Do not change any other parameters. |
| Services | Service is an abstracted way to define a logical set of Pods and how to access them.<br>You need to create services for TIBCO MDM UI, TIBCO MDM Configurator UI, Apache ActiveMQ UI, Apache Ignite, TIBCO Patterns-Search, and Apache ActiveMQ.<br>You must create headless services for the the following components:<br>• TIBCO MDM<br>• TIBCO MDM Configurator<br>• Apache Ignite<br>• TIBCO Patterns-Search<br>• Apache ActiveMQ |

### ConfigMap YAML File Parameters

You can update the ConfigMap parameters based on the configuration that you are using.

The following table lists the parameters available in the ConfigMap YAML file (`docker\k8s_deployment\config_dep.yaml`), their definitions and example values:

| Parameter Name | Definition |
|---|---|
| `MQ_TROPOS_ENVIRONMENT_ID` | Name of the namespace |

| Parameter Name | Definition |
|---|---|
| `MQ_MDM_DB_TYPE` | Database type<br>Example: `POSTGRESQL`, `ORACLE`, and `SQLSERVER` |
| `MQ_MDM_DB_HOST` | Database server host name<br>Example: `pgsqldb-eks.cx4wjme9qqns.us-west-2.rds.amazonaws.com` |
| `MQ_MDM_DB_PORT` | Database port<br>Example:<br>• `5432` for `POSTGRESQL`<br>• `1521` for `ORACLE`<br>• `1433` for `SQLSERVER` |
| `MQ_MDM_DB_NAME` | Database name<br>Example: `velodb` |
| `MQ_MDM_DB_USETABLESPACES` | Set this value to `true` if the TIBCO MDM database uses undefined table spaces. |
| `MQ_MDM_DB_MIN_CONN_COUNT`, `MQ_MDM_DB_MAX_CONN_COUNT` | Minimum and maximum number of database connections for TIBCO MDM database . |
| `MQ_MDM_FAST_CACHE_ENABLED` | To enable golden record cache, set this value as `true`. Else set it as `false`. |
| `MQ_ACTIVEMQ_COMPONENT_IDS` | The name by which TIBCO MDM server discovers other services<br>Example: `MQ_ACTIVEMQ_COMPONENT_IDS: activemq.`*`namespace`*`.svc.cluster.local` |
| `MQ_PATTERNS_COMPONENT_IDS` | Example: `MQ_PATTERNS_COMPONENT_IDS: patterns.`*`namespace`*`.svc.cluster.local` |
| `MQ_IGNITE_COMPONENT_IDS` | Example: `MQ_IGNITE_COMPONENT_IDS: ignite.`*`namespace`*`.svc.cluster.local` |
| `MQ_MDM_COMPONENT_IDS` | Example: `MQ_MDM_COMPONENT_IDS: mdm.`*`namespace`*`.svc.cluster.local` |
| `MQ_TZ` | Set the time zone before you deploy containers in Kubernetes. All containers including the database must be in the same time zone. By default, the time zone is set to `Asia/Kolkata`. |
| **For TIBCO MDM REST API through Swagger UI** | |
| `SWAGGER_MDM_HOST` | An IP address of the TIBCO MDM server on which you want to try out TIBCO MDM REST APIs. |

| Parameter Name | Definition |
|---|---|
| `SWAGGER_MDM_PORT` | Port of the TIBCO MDM server on which you want to try out TIBCO MDM REST APIs. |

## Deploying TIBCO MDM Cluster on Kubernetes

You can deploy TIBCO MDM Cluster on Kubernetes by using the YAML configuration files, which contain the configuration details for deployment.
For more information about the Kubernetes concepts, see Kubernetes Documentation.

### Procedure

1. Create the following Kubernetes objects that are required for deploying TIBCO MDM Cluster, using the YAML files. These objects include deployment and services for the cluster:

    - Namespace
    - Rolebinding
    - Config Map
    - Secret
    - Deployment objects for TIBCO MDM, TIBCO MDM Configurator, Apache Ignite, TIBCO Pattern-Search, and Apache ActiveMQ
    - Services for TIBCO MDM UI, TIBCO MDM Configurator UI, Apache ActiveMQ UI, and headless services for TIBCO MDM, Apache Ignite, TIBCO Pattern - Search, and Apache ActiveMQ.

    For more information on the YAML file configurations of TIBCO MDM Cluster components, see TIBCO MDM Cluster Components YAML Files.

2. On the command line, type the **`create`** command of `kubectl` utility by using the YAML files to deploy TIBCO MDM Cluster components. The deployment files are available in the `docker\k8s_deployment` folder.

```
kubectl create -f namespace.yaml
kubectl create -f namespace.yaml
kubectl create rolebinding default-viewer --clusterrole=view --
serviceaccount=development:default --namespace=development

kubectl create -f config_dep.yaml
kubectl create -f secrets_dep.yaml

kubectl create -f activemq_dep.yaml
kubectl create -f activemq_ser.yaml
kubectl create -f activemqui_ser.yaml

kubectl create -f ignite_dep.yaml
kubectl create -f ignite_ser.yaml

kubectl create -f mdm_sc.yaml
kubectl create -f mdm_pv.yaml
kubectl create -f mdm_pvc.yaml

kubectl create -f patterns_ser.yaml
kubectl create -f patterns_dep.yaml

kubectl create -f mdm_ser.yaml
kubectl create -f mdm_dep.yaml
kubectl create -f mdmui_ser.yaml

kubectl create -f mdmconfig_ser.yaml
kubectl create -f mdmconfig_dep.yaml
```

⚠ After TIBCO MDM starts, deploy the `mdmconfig` YAML files for the TIBCO MDM Configurator. If you deploy TIBCO MDM and TIBCO MDM Configurator at the same time, the `ConfigLogin.info is not found` issue occurs when you log in to the TIBCO MDM Configurator.

## Accessing TIBCO MDM Cluster UI on Kubernetes

By using the following `get services` commands and URLs, you can get the IP address and port of the Cluster containers:

| Cluster Container | Command | URL |
|---|---|---|
| TIBCO MDM | `kubectl get services mdmui` | http://*IP address of mdmui*:30036/eml/Login |
| TIBCO MDM Configurator | `kubectl get services mdmconfig` | http://*IP address of mdmconfig*:30037/config |
| Apache ActiveMQ | `kubectl get services activemqui` | http://*IP address of activemqui*:30038/activemq |

# TIBCO ActiveSpaces

TIBCO ActiveSpaces is a distributed peer-to-peer in-memory data grid, a form of virtual shared memory that leverages a distributed hash table with configurable replication. It simplifies enterprise application development by providing an easy-to-use interface for data storage and retrieval (data grid), and process coordination (messaging grid).

ActiveSpaces is based on the concept of *tuple spaces*—shared storage areas for collections of relational database rows. ActiveSpaces provides coordination and communication, which enables unrelated programs and processes to get, put, take, browse, and listen from the collective tuple spaces across platforms and architectures.

Use the ActiveSpaces software for distributed caching of TIBCO MDM. It provides the scalability, performance, and high availability for the TIBCO MDM data cache, which is needed to support large data volumes and high data access frequencies.

## ActiveSpaces Concepts and Terminology

To configure TIBCO MDM with ActiveSpaces, you must understand the concepts and terminologies of the ActiveSpaces.

### Agent

The amount of data that can be stored in a space depends on the number of seeding members of that space. It can be necessary to add seeders to a space to scale it up. *AS-Agent* is a prebuilt process that users can run on any host whose sole purpose is to join all distributed spaces in the specified metaspace as a seeder. Agents can also be used to ensure that the desired degree of replication specified for a space can be achieved.

### Leech

A member that joins a space but does not lend any resources to the space, such as memory or processing power. Thus, plays a passive role.

### Peer

A process that has connected to a metaspace and joined a space as either a seeder or a leech.

### Metaspace

A *metaspace* is a logical concept representing two things:

- From a deployment perspective, it represents the cluster of hosts and processes sharing the same metaspace name and set of multicast transport attributes, making a particular instance of an ActiveSpaces deployment. The hosts and processes in a metaspace can work together by joining the same tuple spaces.
- From an administrative point of view, a metaspace is a container for a set of spaces. There are two kinds of spaces contained in a metaspace: *system spaces*, which are defined by ActiveSpaces itself, and *user spaces*, meaning spaces that are defined by a user.

A valid metaspace name must conform to the following rules:

- Cannot start with a '$' or '_'
- Can contain alphanumeric characters and '-' or '_'

The metaspace is the initial handle to ActiveSpaces. An application or member first joins a metaspace, and through it, gets access to other objects and functionality. For TIBCO MDM, the metaspace name is part of the configuration in the Configuration value under the name AS Meta Space Name. The default value is cim_ms.

### Spaces

All records of a certain type are stored in a space. For TIBCO MDM, the spaces are identical to the caches. For each record type, there is a different cache. You can configure the caches in the `CacheConfig.xml` file.

### Seeder

A member that joins a space and lends resources, such as memory and processing power, to the scalability of the space. Thus, plays an active role in maintaining the space by providing CPU and RAM.

In a distributed space, all peers are responsible for seeding certain tuples.

In a non-distributed space, one of the peers is assigned to be the seeder, determined by the ActiveSpaces distribution algorithm.

Ideally, peers are relatively stable, since there is overhead to reorganize the distribution of the tuples among the remaining peers when a peer leaves the space. For this reason, a transient application—one that will leave and join the space frequently—should generally be configured to join the space as a leech, rather than as a peer.

Agents are always seeders, not leeches. Agents provide an efficient, stable means of increasing the scalability of a space. Also, multiple seeders cannot be created from a single client program.

For each entry in a space, the ActiveSpaces distribution algorithm designates one seeder as the seeder of that tuple, whether or not the tuple is replicated on other members. The seeder holds and owns the authoritative copy of the complete tuple.

If the space has multiple seeders, a tuple may be held by different seeders at different times. If the current seeder of the entry leaves the space, another seeder is chosen as the entry’s new seeder, and the entry is then copied over to the new seeder.

### Transport URLs

Transport arguments are specified in the form of URLs, one for the multicast transport and the other for unicast transport.

Multicast URL — com.tibco.cim.cache.as.multicasturl

Unicast URL — com.tibco.cim.cache.as.unicasturl

The *multicast URL* is a string specifying the arguments that is used by the multicast transport of ActiveSpaces to discover and communicate with the other members of the metaspace. All intended members of a metaspace must specify compatible multicast URLs in order for them to become members of the same metaspace.

The tibpgm:// and tcp:// URLs should be sufficient in most cases (single server and single subnetwork). The complete syntax is listed in the subsequent sections.

### PGM (Pragmatic General Multicast) URL Format

The following multicast URL format means that the PGM multicast transport is used:

```
tibpgm://[dport]/[interface]/[discovery IP multicast address]/[option=value;]*
```

- `[dport]` specifies the destination port used by the PGM transport.

  > You must specify the unique port number in the Value attribute, else an error message is displayed. For example, 12345.

- `[interface]/[discovery IP multicast address]` specifies the address of the interface to be used for sending multicast packets, and the multicast group address to be used. If not specified, the default interface and multicast address is used, that is, `239.8.8.8`.

- `[option=value;]` is a semicolon-separated list of optional PGM transport arguments. For example:
  - source_max_trans_rate=100000000 (in bits per second) confines the PGM transport to limit its transmission rate to 100 megabits per second.
  - By default, the PGM transport is tuned to provide the best performance according to the most common deployment architectures, and the values of those optional arguments should only be changed when necessary, and with care, as inappropriate values could easily result in degraded performance of the product.

    Creating raw PGM packets (as opposed to UDP-encapsulated PGM packets) requires the process to have root privileges on UNIX-based systems.

### Unicast URL Format

To use a unicast URL, use a string of the following form:

```
   tcp://interface/port
```

This syntax indicates that the member should bind to the specified interface and the specified port when creating the TCP socket that will be used for direct communication between the members of the metaspace. If not specified, it will default to `0.0.0.0` (INADDR_ANY) for the interface and `0` (any free port) for the port.

A successful connection to the metaspace will return a valid instance of a Metaspace object, which can then be used to define, join, or leave spaces.

### Unicast Discovery

If unicast discovery is enabled, every instance of ActiveSpaces tries to connect to a node in the discovery URL list in the order they are provided. Once connected to any of them, the node get information about “current manager” in the metaspace, and continue operations. If this node is not able to connect any member in the discovery URL list, the node indicates itself as a manager.

In Configurator, you can specify the discovery URL (com.tibco.cim.cache.as.discoveryurl) in the following format: IP-Address:port number.

### Listen URL

By default, an ActiveSpaces instance listens on a random port in 30K+ range for ActiveSpaces internal data communication. If this range of ports are blocked, then provide an available port in Configurator (com.tibco.cim.cache.as.listenurl) in the following format: tcp://IP_address:port number.

To start an agent, use the following command:

**`as-agent.exe -metaspace="cim_ms" -discovery ""tcp://IP_A:7888""`** where IP_A is the self IP of the local machine where you are starting agent.

In Configurator, add the following IP discovery address to detect the agent: `IP_A:7888`.

since two or more nodes share Configurator in a cluster, specify a port number for the listen URL, but not a specificIP address. For example,

Instead of

```
<ConfValue description="Provide the listen url for AS" name="AS Listen URL"
propname="com.tibco.cim.cache.as.listenurl" sinceVersion="8.2"
visibility="Advanced">        <ConfString default="tcp://"  value="tcp://
10.254.6.109:50001"/>   </ConfValue>
```

To

```
<ConfValue description="Provide the listen url for AS" name="AS Listen URL"
propname="com.tibco.cim.cache.as.listenurl" sinceVersion="8.2"
visibility="Advanced">        <ConfString default="tcp://"  value="tcp://:50001"/
>   </ConfValue>
```

In this example, port `50001` will be used on all nodes of the cluster.

### Tuples

A *tuple* is similar to a row in the database table. Specifically, it is a sequence of named elements called fields (similar to the columns in a database table) which contain values of a specific type. Each tuple in a space represents a set of related data. Fields have a name and a type. A tuple can be seen as a kind of map on which fields can be *put* or *removed*.

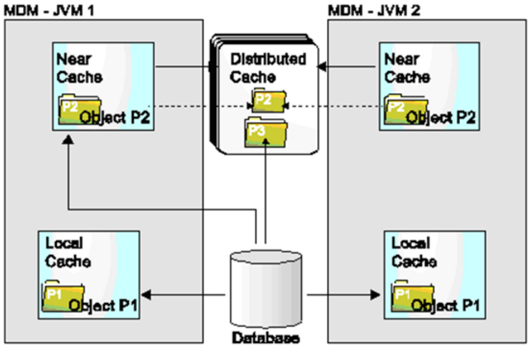
## Caching Engine

TIBCO MDM implements a caching engine using TIBCO ActiveSpaces. To optimize the cache latency and network load, caching engine implements three types of caches.

- **Local Cache** – The Local cache is an on-heap cache, designed for caching objects. The objects are not required to be synchronized across other servers. The Local cache is used for frequently updated objects that are updated only on one node and are not required by other nodes in the cluster.
- **Near Cache** – The Near cache is an on-heap cache that stores data. The data does not change often and does not require a lot of memory. A change to the data results into an update to all other copies of that data stored on other nodes. The Near cache is used for infrequently updated objects that are read very frequently. The Near cache provides optimal read time without a network hop and the overhead of data deserialization.
- **Distributed Cache** – The Distributed cache implements high performance shared distributed cache. It is a non-transactional side cache. The Distributed cache is used for objects that can be updated and read from any node and which require fault tolerance. A single cache is accessed from all the nodes in the cluster.

  When TIBCO MDM retrieves any data, it searches data in the cache. If the data is not found in cache, data is loaded from the database into the cache. However, a warm caching can be configured to preload most of the required data at startup. Depending on the available memory, all or part of the data can be quickly loaded using multithreaded loading. After the data is loaded in cache, TIBCO MDM attempts to retain it in cache and update it when data is changed.

*Different Type of Caching with TIBCO MDM*



TIBCO MDM uses the distributed cache to keep the master data in memory for faster manipulation. The distributed cache stores data, which is already committed to the database. The primary driver reduces the number of hits to database for data reads. The distributed cache also implements synchronization process across the TIBCO MDM cluster.

TIBCO MDM embeds TIBCO ActiveSpaces to implement the distributed cache and supports various configurations of ActiveSapces, that is, replication, eviction policies, and so on.

## Distributed Cache

You can configure the distributed cache for failover by replicating cached objects. This is only mandatory for a small number of objects, which are not persisted to database. However, other objects stored in the Distributed cache can also be replicated so that loss of a cache node does not require data reload.

Other cache usages include:

- Distributed lock to synchronize data across multiple instances
- Remote invocation of object methods
- Temporary store for ephemeral objects, that is, an intermediate workflow state and work files

# Configuring TIBCO ActiveSpaces

Configure ActiveSpaces on your computer to work with TIBCO MDM.

### Procedure

1. Create a system environment variable called AS_HOME that points to the directory where ActiveSpaces is installed, for example, `$MQ_HOME/bin/as/`*version*.
2. Add the following system properties on Computers where ActiveSpaces and TIBCO MDM are installed:

   -
     ```
     <property name="PATH" value="C:/Apps/tibco/mdm/version/bin/as/version/bin;C:/
     Apps/tibco/mdm/version/bin/as/version/lib;${PATH}"/>
     ```

     The `PATH` system property is applicable only for Windows and Linux.

   -
     ```
     <property name="LD_LIBRARY_PATH" value=" /Apps/tibco/mdm/version/bin/as/bin:/
     Apps/tibco/mdm/version/
     bin/as/version/lib:${ LD_LIBRARY_PATH}"/>
     ```

     The LD_LIBRARY_PATH system property is applicable only for non-Windows environment, that is, Linux.

# Topologies to Configure TIBCO MDM with ActiveSpaces

You can configure TIBCO MDM with ActiveSpaces using one of the cluster topologies.

- **Single Server Embedded Cache**: refer to Single Server Embedded Cache Configuration
- **Peer-to-Peer Server**: refer to Peer-to-Peer Server Configuration
- **Centralized Cache Server**: refer to Centralized Cache Server Configuration

## Cache Configuration Properties for Topology

The Cache Configuration Properties for Topology table lists the cache configuration properties that are required for each topology.

*Cache Configuration Properties for Topology*

| Property Name | Embedded Cache | Peer-to-Peer Server | Centralized Cache Server |
|---|---|---|---|
| **M** = Mandatory, **O** = indicates Optional, and **N** = Not Required | | | |

| Property Name | Embedded Cache | Peer-to-Peer Server | Centralized Cache Server |
|---|---|---|---|
| AS Meta Space Name Type (com.tibco.cim.cache.as.metaspace ) | M | M | M |
| AS unicast discovery URLs (com.tibco.cim.cache.as.discoveryurl) | N | N | M |
| AS Listen URL (com.tibco.cim.cache.as.listenurl )<br>(Optional) | O | O | M |
| AS Unicast URL (com.tibco.cim.cache.as.unicasturl ) | N | N | N |
| AS Multicast URL (com.tibco.cim.cache.as.multicasturl ) | M | M | N |
| AS Member Distribution Role (com.tibco.cim.cache.as.distributionrole) | M | M | M |

## Cluster and Member-Specific Properties

The Cluster and Member-Specific Properties table lists the properties that support cluster and member.

*Cluster and Member-Specific Properties*

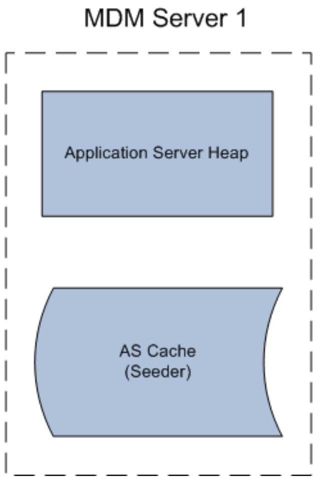
| Property Name | Cluster Specific | Member Specific |
|---|---|---|
| AS Meta Space Name Type (com.tibco.cim.cache.as.metaspace ) | a | r |
| AS unicast discovery URLs (com.tibco.cim.cache.as.discoveryurl) | a | r |
| AS Listen URL (com.tibco.cim.cache.as.listenurl )<br>(Optional) | r | a |
| AS Unicast URL (com.tibco.cim.cache.as.unicasturl ) | r | r |
| AS Multicast URL (com.tibco.cim.cache.as.multicasturl ) | a | r |
| AS Member Distribution Role (com.tibco.cim.cache.as.distributionrole) | r | a |

## Single Server Embedded Cache Configuration

In the Single Server Embedded Cache topology, a single TIBCO MDM node itself contains ActiveSpaces. You do not need to run the external cache server.

For example:

*Single Server Embedded Cache*



To configure Single Server, specify the following properties in Configurator.

*Single Server Embedded Cache Configuration*

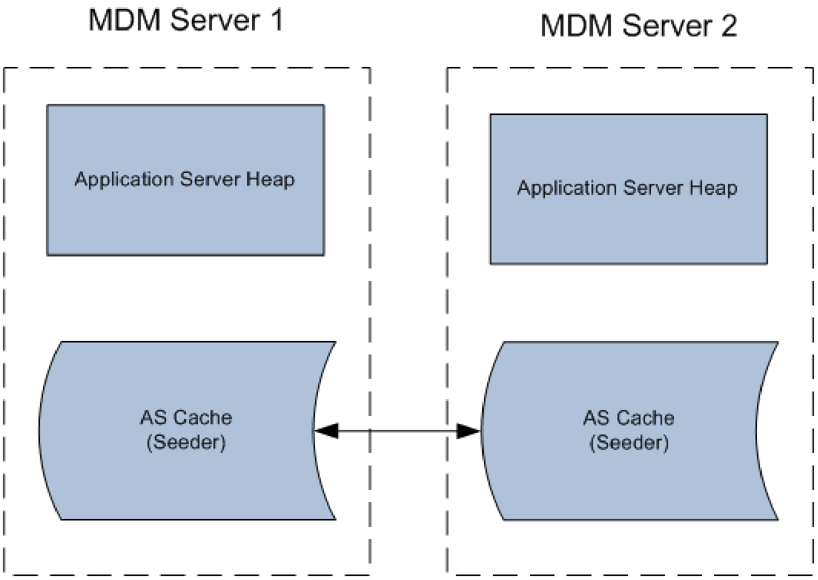
| Property Name | Description | Values |
|---|---|---|
| AS Meta Space Name Type (com.tibco.cim.cache.as.metaspace ) | Specify the metaspace name. | For example, `cim_ms_prod`. |
| AS unicast discovery URLs (com.tibco.cim.cache.as.discoveryurl) | Not Required. | Not Required. |
| AS Listen URL (com.tibco.cim.cache.as.listenurl )<br>(Optional) | Specify the listen URL. | For example, `tcp://10.20.30.41:50001` indicates to use `50001` on the member's IP: 10.20.30.41. |
| AS Unicast URL (com.tibco.cim.cache.as.unicasturl ) | Specify the unicast URL or use the default. | The default value is `tcp:////`. |
| AS Multicast URL (com.tibco.cim.cache.as.multicasturl ) | Specify the multicast URL or use the default. | For example, `tibpgm://12348` |
| AS Member Distribution Role (com.tibco.cim.cache.as.distributionrole) | Select `SEEDER` from the drop-down list. | The default value is `SEEDER`. |

## Peer-to-Peer Server Configuration

In the Peer-to-Peer Server topology, two or more TIBCO MDM server runs as AS Seeder. The TIBCO MDM server starts the embedded ActiveSpace seeder server. All processes or nodes are direct peers to each other and there are no *servers* or *clients*, rather seeders (contributing nodes) and leeches (non-contributing nodes).

For example:

***Peer-to-Peer Server***



While setting up the Peer-to-Peer Server architecture, specify all seeders with the ActiveSpaces configuration.

- In the case of one seeder and multiple leeches, only the seeder needs the ActiveSpace configuration.
- In all cases, all nodes need the metaspace name, listen URL, and so on to set up the communication channels with one another.

You can configure a clustered TIBCO MDM ActiveSpaces instance using the following two methods:

- Peer-to-Peer Server Configuration Using Multicast URL
- Peer-to-Peer Server Configuration Using Discovery URL

## Peer-to-Peer Server Configuration Using Multicast URL

Before peer-to-peer server configuration using multicast URL, ensure that your system meets all of the requirements.

Ensure the following requirements:

- Metaspace name and Multicast URL must be the same for all nodes.
- Any port, supported by ActiveSpaces can be used for these members. If two members are on the same host or IP address, the port must be different for both the members.
- Any member can use any distribution role. However, at least one SEEDER must be available and that should start first.

*Peer-to-Peer Server Configuration Using Multicast URL*

| Property Name | Member1 | Member2 | Member3 |
|---|---|---|---|
| AS Meta Space Name Type (com.tibco.cim.cache.as. metaspace ) | `For example, cim_ms_prod` | `For example, cim_ms_prod` | `For example, cim_ms_prod` |
| AS unicast discovery URLs (com.tibco.cim.cache.as. discoveryurl) | Not Required | Not Required | Not Required |
| AS Listen URL (com.tibco.cim.cache.as.l istenurl )<br>(Optional) | For example, `tcp://10.20.30.41:50000` | For example, `tcp://10.20.30.42:50000` | For example, `tcp://10.20.30.43:50000` |
| AS Unicast URL (com.tibco.cim.cache.as. unicasturl ) | Not Required | Not Required | Not Required |
| AS Multicast URL (com.tibco.cim.cache.as. multicasturl ) | For example, `tibpgm://12348` | For example, `tibpgm://12348` | For example, `tibpgm://12348` |
| AS Member Distribution Role (com.tibco.cim.cache.as. distributionrole) | Select SEEDER from the drop-down list. | Select LEECH from the drop-down list. | Select SEEDER from the drop-down list. |

## Peer-to-Peer Server Configuration Using Discovery URL

Before peer-to-peer server configuration using discovery URL, ensure that your system meets all of the requirements.

Ensure the following requirements:

- Metaspace name and Discovery URL List contain listen URL of all nodes.
- Any port, supported by ActiveSpaces can be used for these members. If two members are on the same host or IP address, the port must be different for both.
- Any member can use any distribution role. However, at least one SEEDER must be available and that should start first.

*Peer-to-Peer Server Configuration Using Discovery URL*

| Property Name | Member1 | Member2 | Member3 |
|---|---|---|---|
| AS Meta Space Name Type (com.tibco.cim.cache.as. metaspace ) | `For example, cim_ms_prod` | `For example, cim_ms_prod` | `For example, cim_ms_prod` |

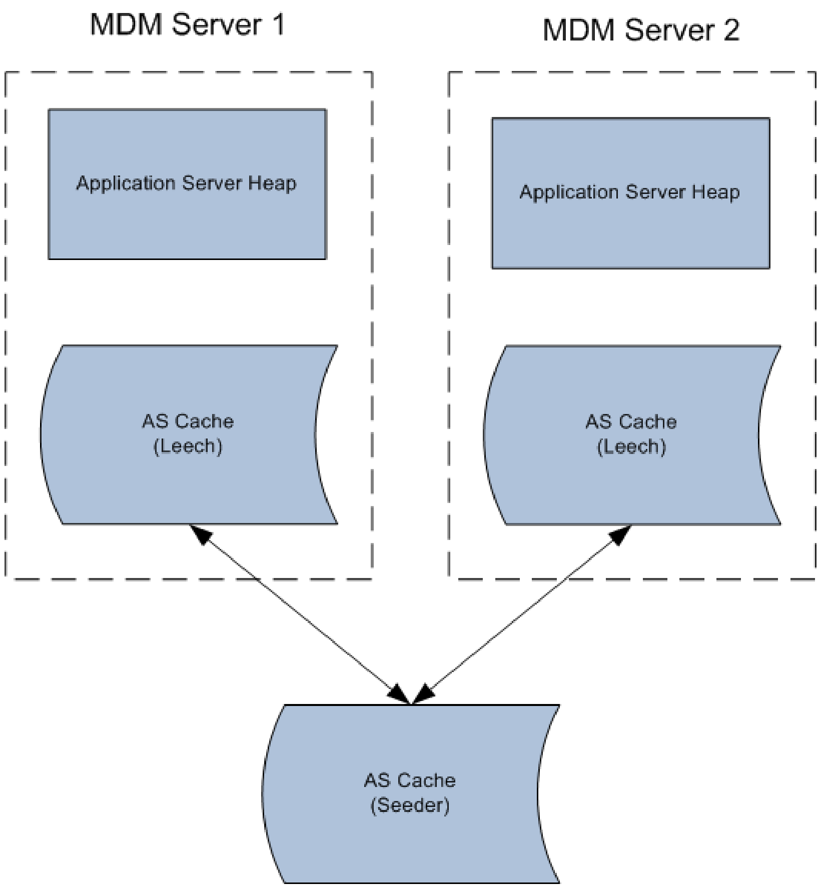
| Property Name | Member1 | Member2 | Member3 |
|---|---|---|---|
| AS unicast discovery URLs (com.tibco.cim.cache.as. discoveryurl) | For example, `tcp://10.20.30.41:50000`<br>`tcp://10.20.30.42:50000 tcp://10.20.30.43:50000` | For example, `tcp://10.20.30.41:50000`<br>`tcp://10.20.30.42:50000 tcp://10.20.30.43:50000` | For example, `tcp://10.20.30.41:50000`<br>`tcp://10.20.30.42:50000 tcp://10.20.30.43:50000` |
| AS Listen URL (com.tibco.cim.cache.as.listenurl )<br>(Optional) | For example, `tcp://10.20.30.41:50000` | For example,<br>`tcp://10.20.30.42:50000` | For example,<br>`tcp://10.20.30.43:50000` |
| AS Unicast URL (com.tibco.cim.cache.as. unicasturl ) | | | |
| AS Multicast URL (com.tibco.cim.cache.as. multicasturl ) | | | |
| AS Member Distribution Role (com.tibco.cim.cache.as. distributionrole) | Select `SEEDER` from the drop-down list. | Select `LEECH` from the drop-down list. | Select `SEEDER` from the drop-down list. |

## Centralized Cache Server Configuration

In the Centralized Cache Server topology, two or more TIBCO MDM servers run with ActiveSpaces LEECH, and a dedicated AS-Agent runs as a Seeder. For example:

For example:

*Centralized Cache Server*



Ensure the following:

- Metaspace name and Discovery URL List contain listen URL of all nodes.
- Any port, supported by ActiveSpaces can be used for these members. If two members are on the same host or IP address, the port must be different for both the members.
- Any member can use any distribution role. However, at least one SEEDER that starts first should be available.

To configure Centralized Cache Server, specify the following properties in Configurator:

*Centralized Cache Server Configuration*

| Property Name | Member1 | Member2 | AS-Agent1 | AS-Agent2 |
|---|---|---|---|---|
| AS Meta Space Name Type (com.tibco.cim.cache.as.metaspace) | `For example, cim_ms_prod` | `For example, cim_ms_prod` | `For example, cim_ms_prod` | For example, `cim_ms_prod` |

| Property Name | Member1 | Member2 | AS-Agent1 | AS-Agent2 |
|---|---|---|---|---|
| AS unicast discovery URLs (com.tibco.cim.cache.as.discoveryurl) | For example, `tcp://10.20.30.41:50000`<br>`tcp://10.20.30.42:50000`<br>`tcp://10.20.30.43:50000`<br>`tcp://10.20.30.44:50000` | For example, `tcp://10.20.30.41:50000`<br>`tcp://10.20.30.42:50000`<br>`tcp://10.20.30.43:50000`<br>`tcp://10.20.30.44:50000` | For example, `tcp://10.20.30.41:50000`<br>`tcp://10.20.30.42:50000`<br>`tcp://10.20.30.43:50000`<br>`tcp://10.20.30.44:50000` | For example, `tcp://10.20.30.41:50000`<br>`tcp://10.20.30.42:50000`<br>`tcp://10.20.30.43:50000`<br>`tcp://10.20.30.44:50000` |
| AS Listen URL (com.tibco.cim.cache.as.listenurl)<br>(Optional) | For example, `tcp://10.20.30.41:50000` | For example,<br>`tcp://10.20.30.42:50000` | For example,<br>`tcp://10.20.30.43:50000` | For example,<br>`tcp://10.20.30.44:50000` |
| AS Unicast URL (com.tibco.cim.cache.as.unicasturl ) | | | | |
| AS Multicast URL (com.tibco.cim.cache.as.multicasturl) | | | | |
| AS Member Distribution Role (com.tibco.cim.cache.as.distributionrole) | Select `LEECH` from the drop-down list. | Select `LEECH` from the drop-down list. | `SEEDER` | `SEEDER` |

## Starting AS-Agent

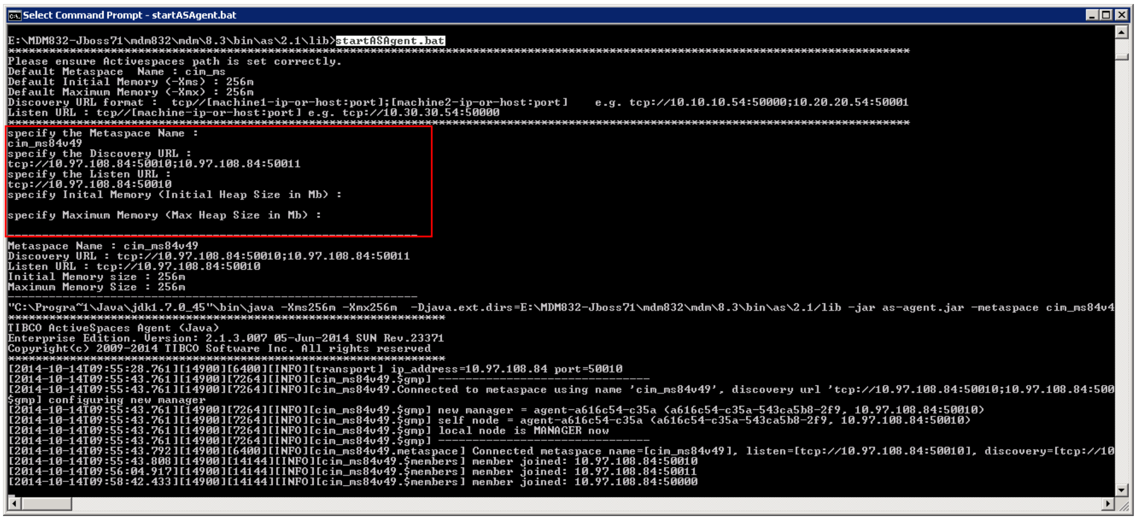
Start the AS-Agent by running the `startASAgent.bat` file.

### Prerequisites

Before starting the AS-Agent, copy `ECMClasses.jar` from `$MQ_HOME/lib/mq` to `$AS_HOME/lib`. By default, the `cim_ms` is defined for the `Metaspace Name` property in Configurator and `512` is defined for the Memory attribute in the `CacheConfig.xml` file. You can change these values.

### Procedure

1. To start AS-Agent, run the `startASAgent.bat` file located at `$MQ_HOME/bin`.



```
E:\MDM832-Jboss71\mdm832\mdm\8.3\bin\as\2.1\lib>startASAgent.bat
****************************************************************************************************************
Please ensure Activespaces path is set correctly.
Default Metaspace  Name : cim_ms
Default Initial Memory (-Xms) : 256m
Default Maximum Memory (-Xmx) : 256m
Discovery URL format :  tcp//[machine1-ip-or-host:port];[machine2-ip-or-host:port]    e.g. tcp://10.10.10.54:50000;10.20.20.54:50001
Listen URL : tcp//[machine-ip-or-host:port] e.g. tcp://10.30.30.54:50000
****************************************************************************************************************
specify the Metaspace Name :
cim_ms84v49
specify the Discovery URL :
tcp://10.97.108.84:50010;10.97.108.84:50011
specify the Listen URL :
tcp://10.97.108.84:50010
specify Inital Memory (Initial Heap Size in Mb) :

specify Maximum Memory (Max Heap Size in Mb) :

------------------------------------------------------------
Metaspace Name : cim_ms84v49
Discovery URL : tcp://10.97.108.84:50010;10.97.108.84:50011
Listen URL : tcp://10.97.108.84:50010
Initial Memory size : 256m
Maximum Memory Size : 256m
------------------------------------------------------------
"C:\Progra~1\Java\jdk1.7.0_45"\bin\java -Xms256m -Xmx256m  -Djava.ext.dirs=E:\MDM832-Jboss71\mdm832\mdm\8.3\bin\as\2.1/lib -jar as-agent.jar -metaspace cim_ms84v4
*************************************************************
TIBCO ActiveSpaces Agent (Java)
Enterprise Edition. Version: 2.1.3.007 05-Jun-2014 SUN Rev.23371
Copyright(c) 2009-2014 TIBCO Software Inc. All rights reserved
*************************************************************
[2014-10-14T09:55:28.761][14900][6400][INFO][transport] ip_address=10.97.108.84 port=50010
[2014-10-14T09:55:43.761][14900][7264][INFO][cim_ms84v49.$gmp] ------------------------------
[2014-10-14T09:55:43.761][14900][7264][INFO][cim_ms84v49.Connected to metaspace using name 'cim_ms84v49', discovery url 'tcp://10.97.108.84:50010;10.97.108.84:500
$gmp] configuring new manager
[2014-10-14T09:55:43.761][14900][7264][INFO][cim_ms84v49.$gmp] new manager = agent-a616c54-c35a (a616c54-c35a-543ca5b8-2f9, 10.97.108.84:50010)
[2014-10-14T09:55:43.761][14900][7264][INFO][cim_ms84v49.$gmp] self node = agent-a616c54-c35a (a616c54-c35a-543ca5b8-2f9, 10.97.108.84:50010)
[2014-10-14T09:55:43.761][14900][7264][INFO][cim_ms84v49.$gmp] local node is MANAGER now
[2014-10-14T09:55:43.761][14900][7264][INFO][cim_ms84v49.$gmp] ------------------------------
[2014-10-14T09:55:43.792][14900][6400][INFO][cim_ms84v49.metaspace] Connected metaspace name=[cim_ms84v49], listen=[tcp://10.97.108.84:50010], discovery=[tcp://10
[2014-10-14T09:55:43.808][14900][14144][INFO][cim_ms84v49.$members] member joined: 10.97.108.84:50010
[2014-10-14T09:56:04.917][14900][14144][INFO][cim_ms84v49.$members] member joined: 10.97.108.84:50011
[2014-10-14T09:58:42.433][14900][14144][INFO][cim_ms84v49.$members] member joined: 10.97.108.84:50000
```

2. Based on the specified parameters in the `startASAgent.bat` file, you are prompted to enter the values of environment variables. For example,

   - the Metaspace Name
   - the Discovery URL
   - the Listen URL

3. Invoke the following command from the `$MQ_HOME/bin` directory to display the **help** for AS-Agent:

   ```
   C:\tibco\mdm\version\bin\as\version\lib>java -jar as-agent.jar
   -help
   ```

## Result

The following is the output of the Help request. If you do not specify any parameter, it includes the default values.

```
Usage
      -metaspace <metaspace_name>  default ms
      -multicast <multicast_url>   default tibrv://
      -unicast <unicast_url>       default tcp://
      -log <log_file>
          -debug <log_level>           default 3 (INFO)
Multicast url format:
      tibpgm://dport/interface;multicast/key1=value1;key2=value2;
         key3=value3
      tibrv://service/network/daemon
Unicast url format:
      tcp://interface/listen port
```

## AS-Agent Parameters

The AS-Agent Parameters table describes the AS-Agent parameters and their usage.

*As-Agent Parameters*

| Parameter Name | Sample URL | Description |
|---|---|---|
| Multicast URL Format | `tibpgm://[dport]/[interface]/[discovery IP multicast address]/[option=value;]*` | • dport  indicates destination port<br>• interface  indicates the IP address<br>• multicast  indicates the multicast IP address<br>• option=value indicates the option user property has value as its value. These parameters must be known to PGM; otherwise, they are ignored. |
| Unicast URL Format | `tcp://`*`interface`*`: `*`listen port`* | • interface: on Windows, the interface can be an IP address or an interface number in a format like IP00 or IP01. For example:<br>`tcp://192.168.1.191/` or `tcp://IP00/` or `tcp://IP01/`<br>On Linux, the interface can be an IP address or interface number in a format like eth0 or eth1. For example:<br>`tcp://192.168.1.191/` or `tcp://eth0/` or `tcp://eth1/`<br>• listen port: The listen port can be a user-defined free port on the system. If nothing is specified, then the system chooses an available free port. |
| Log | `-log` | If the `-log` as parameter is used, then the output is `as-<processid>.log`. |
| -debug <log_level> | `- debug `*`log level`* | The default is 3 (INFO). The log information displayed on the console is minimal and cannot be controlled through this parameter. This parameter is only for log files. If a log file is not specified, then the debug (log level) value is ignored |

| Parameter Name | Sample URL | Description |
|---|---|---|
| Security Policy File Path (For ActiveSpaces Security) | `security policy file path -` *`security_policy`* | Refers to the path of a security policy file.<br>This parameter is mandatory when ActiveSpaces security is applied and if the `startAgent` utility works as a `SECURITY_DOMAIN_CONTROLLER`. |
| Security Token File Path (For ActiveSpaces Security) | `security token file path -` *`security_token`* | Refers to the path of a security token file.<br>This parameter is mandatory when ActiveSpaces security is applied and the `startAgent` utility works as a `SECURITY_DOMAIN_REQUESTOR`. |

## ActiveSpaces Security

Using the ActiveSpaces security, you can restrict connecting any unauthorized external node to another node if metaspace name or discovery URL is known. You can also restrict to reading, writing, deleting, or altering the data without any constraint. The following features of ActiveSpaces security are implemented in TIBCO MDM:

- Security domain: two types of nodes in the security domain are security domain controller and security domain requestor.
- Security domain controller: enforces the defined behavior of the security domain for a metaspace.
- Security domain requestor: nodes in the metaspace that request secured services from the security domain controllers are called security domain requestors. For example, nodes that require access to the data in the data grid, such as a seeder or a leech, and which must be authorized by a controller.

For more information on these features, refer to *TIBCO ActiveSpaces Developer's Guide*.

### Configuration Properties for ActiveSpaces Security

*Configuration Properties for ActiveSpaces Security*

The following table lists the properties of ActiveSpaces Security specified in Configurator:

| Property Name | Location | Value | Description |
|---|---|---|---|
| Enable activespaces security (`com.tibco.mdm.cache.as.security.enabled`) | Cluster level (InitialConfig > ActiveSpaces security configuration) | `True` or `False`. The default value is False. | Enables the security of ActiveSpaces. |

| Property Name | Location | Value | Description |
|---|---|---|---|
| ActiveSpaces security implementation class (`com.tibco.mdm.cache.as.security.implementationclass`) | Cluster level (InitialConfig > ActiveSpaces security configuration) | The default value is `com.tibco.mdm.infrastructure.cache.as.security.MqActiveSpacesSecurity`. | Refers to the ActiveSpaces security implementation class, which is responsible for retrieving the required parameters for ActiveSpaces security. |
| ActiveSpaces security authentication (`com.tibco.mdm.cache.as.requesternode.authentication.type`) | Cluster level (InitialConfig > ActiveSpaces security configuration) | The supported values are `USERPWD`, `X509V3`, and `NONE`. The default value is `NONE`. | Indicates the authentication type if it is enabled in a policy file. |
| Activespaces authentication callback class name (`com.tibco.mdm.cache.as.security.authentication.class`) | Cluster level (InitialConfig > ActiveSpaces security configuration) | The default value is `com.tibco.mdm.infrastructure.cache.as.security.MqActiveSpacesDefaultAuthenticator`. | Using TIBCO ActiveSpaces security, you can authenticate the users of security domain requestor. The value of this property is the callback class called by ActiveSpaces, when authentication is enabled in a policy file and when the authentication type is set to `USERPWD` or `X509V3`. Otherwise, ActiveSpaces prompts the credentials on the application console. |
| AS Security Domain node type (`com.tibco.mdm.cache.as.security.domain.nodetype`) | Node level (Member1 > Optimization > Member) | The supported values are `SECURITY_DOMAIN_CONTROLLER` and `SECURITY_DOMAIN_REQUESTOR`. The default value is `SECURITY_DOMAIN_CONTROLLER`. | Specifies whether the node works as the domain controller or domain requestor. |

| Property Name | Location | Value | Description |
|---|---|---|---|
| AS Security Network domain name (`com.tibco.mdm.cache.as.requesternode.os.networkdomain`) | Node level (Member1 > Optimization > Member) | Any valid domain name. The default value is null. | Refers to the login domain name of the logged-in user, who is working as a domain requestor. Used if you have specified authentication type as `USERPWD`. This property used when node is working as a domain requestor. |
| AS Security Logged in operating system username of requester node (`com.tibco.mdm.cache.as.requesternode.os.username`) | Node level (Member1 > Optimization > Member) | Any valid user name. The default value is null. | Refers to the login user name of the logged-in user, who is working as a domain requestor. Used if you have specified the authentication type as `USERPWD`. This property is used when the node is working as a domain requestor. |
| AS Security Encrypted password of logged in user of requester node (`com.tibco.mdm.cache.as.requesternode.os.password`) | Node level (Member1 > Optimization > Member) | Any valid password. The default value is null. | Refers to the login password of the logged-in user, who is working as a domain requestor. Used if you have specified the authentication type as `USERPWD`. This property is used when node is working as a domain requestor. Password must be encrypted. |
| AS Security Encrypted Domain or token password (`com.tibco.mdm.cache.as.securitydomain.identity.password`) | Node level (Member1 > Optimization > Member) | The default value is null. | Indicates the password of the domain if the node is working as a domain controller or token password if node is working as a domain requester. |
| AS Security Encrypted password of x509v3 system authentication (`com.tibco.mdm.cache.as.requesternode.x509v3.password`) | Node level (Member1 > Optimization > Member) | The default value is null. | Certificate encrypted password for the `x509v3` type of authentication. This property used when the node is working as a domain requestor. |

| Property Name | Location | Value | Description |
|---|---|---|---|
| AS Security Encrypted password of x509v3 system authentication (com.tibco.mdm.cache.as.requesternode.x509v3.keyfile) | Node level (Member1 > Optimization > Member) | The default value is null. | The key file path of the x509v3 certificate when the x509v3 system authentication is enabled in policy. This property is used when the node works as a domain requestor. |

## Environment Variables for ActiveSpaces Security

You need to specify environment variables for the security policy and token files. These files contain all certificate-related information. The policy file contains the security settings for one or more security domains. The security token file is generated from a security policy file. The Metaspace Access List for the specified security domain is copied from the security policy file into the security token file. The security token file connects to the metaspace by a domain requestor.

Specify the following environment variables in the JVM parameters of the application server that is used to deploy TIBCO MDM. For example, for the JBoss WildFly application server, add environment variables in the standalone.xml file and for the WebSphere and WebLogic application servers, log in to the Administrative console and add the environment variables.

| Environment Variable | Value | Description |
|---|---|---|
| AS_SECURITY_POLICY | Refers to the path of a security policy file. | Required if you have specified the default value, SECURITY_DOMAIN_CONTROLLER of the AS Security Domain node type property for the MDM node. |
| AS_SECURITY_TOKEN | Refers to the path of a security token file. | Required if you have specified the value, SECURITY_DOMAIN_REQUESTOR of the AS Security Domain node type property for the MDM node. |

For creating and validating the security policy and token files, refer to the chapter, " Administering ActiveSpaces with the Admin CLI" in *TIBCO ActiveSpaces Administration*.

## Starting MDM Node as Domain Controller

The MDM node can work as a security domain controller.

### Procedure

1. Add the metaspace name and discovery nodes in a policy file that you plan to use in your application.
2. Add the discovery nodes of all listen URLs, which work as a domain controller in the discovery URL list.
3. Set the com.tibco.mdm.cache.as.security.domain.nodetype property to SECURITY_DOMAIN_CONTROLLER in Configurator.
4. Set AS_SECURITY_POLICY environment variable in the application server environment and specify the path of a policy file.

5. Set the com.tibco.mdm.cache.as.securitydomain.identity.password to the password of your created domain in Configurator.
6. If you have enabled the ActiveSpaces security authentication, specify the implementation class for the ActiveSpaces security implementation class property and user name for the AS Security Logged in operating system username of requester node property. If authentication is not enabled, you must enter the credentials at the application server console.

## Starting MDM Node as Domain Requestor

The MDM node can work as a security domain requestor.

### Procedure

1. Add the metaspace name and discovery nodes in a token file that you connect. You can copy both from the policy file.
2. Set the com.tibco.mdm.cache.as.security.domain.nodetype property to SECURITY_DOMAIN_REQUESTOR in Configurator.
3. Set the AS_SECURITY_TOKEN variable in the application server environment and specify the path of a token file.
4. Set com.tibco.mdm.cache.as.securitydomain.identity.password to the password of your created token in Configurator. It is a token password.
5. If you have enabled the ActiveSpaces security authentication, specify the implementation class for the ActiveSpaces security implementation class property and set the values for the following properties according to the authentication type in Configurator:
    - com.tibco.mdm.cache.as.requesternode.os.networkdomain
    - com.tibco.mdm.cache.as.requesternode.os.username
    - com.tibco.mdm.cache.as.securitydomain.identity.password
    - com.tibco.mdm.cache.as.requesternode.x509v3.password
    - com.tibco.mdm.cache.as.requesternode.x509v3.keyfile

# Cache Configuration

The CacheConfig.xml file includes the cache configuration. The file is available in the $MQ_HOME/config folder.

The following table describes the Cache attributes listed in the CacheConfig.xml file:

*Cache Configuration Parameters*

| Cache Attributes | Descriptions |
|---|---|
| **Server Config** | |
| CacheServerCount | Defines the number of configured cache servers. However, this number is only for the information purpose. |

| Cache Attributes | Descriptions |
|---|---|
| CharSet | Defines the objectsize configuration to be used. The available charsets are singlebyte and multibyte.<br>If the language that is used in the application contains multi byte characters, specify the multibyte value. For example, multi byte characters are used in French, Japanese, Korean, and so on. |
| Memory | Defines the total memory assigned to the cache servers. This includes the heap storage. The memory allocated for all near and distributed caches is (Memory - HeapStorage).<br>Memory must be more than heap size. A warning is issued if memory is specified less than 64m or Memory - Heapstorage is less than 64m. By default, memory is set to 512m.<br>The memory required by external cache servers relies on configuration specified in the `CacheConfig.xml` file. If the memory is 2048 KB and the heapsize is 512 KB, the memory used by each cache server is as follows: (Memory - HeapStorage) 2048 - 512= `1536 KB`. To allocate more memory, you must start more cache servers.<br>You can preload entire repositories into memory at start up. This takes a while to start. However, after the memory is loaded, it provides significant performance gains. Do not use multivalue attributes that are not stored in the shared tables because preloading such large data takes significantly longer. |
| HeapStorage | Defines the memory allocated for near and local caches. Min heap must be minimum 32m, by default, it is 128m. A warning is issued if heap is specified more than 512m. |
| OverHead | Defines the additional overhead added to objectsize. By default, the overhead factor is 1.5. |
| ReplicationCount | Allows you to keep the data multiple times (typically, duplicating it) so that no single server failure can lead to data loss.<br>The value are `0` and `1`. By default, it is set to `0`.<br>• If you select `0`, keeps one copy of the data.<br>• If you select `1`, keeps two copies data records on physically different machines.<br>For more information, refer to Space Replication. |
| **CacheList** | |
| Name | Defines the name of the cache server. For example, RECORD. It allows you to keep a track of the cache configurations. |
| Description | Defines the description of the cache. For example, The description for RECORD is 3-4 entries per record. |

| Cache Attributes | Descriptions |
|---|---|
| Type | Defines the type of cache servers. The available cache server types are local, distributed, and near.<br>Specify limit or list size for near caches to avoid mismatch between capacity in heap and in distributed storage.<br>For distributed caches, if no limit or list size is specified, remaining memory is distributed evenly after memory is assigned to all other caches. The remaining memory must be minimum 32. |
| ReplicationCount | Allows you to keep the data multiple times (typically, duplicating it) so that no single server failure can lead to data loss.<br>The value are `0` and `1`. By default, it is set to `0`.<br>• If you select `0`, keeps one copy of the data.<br>• If you select `1`, keeps two copies data records on physically different machines.<br>For more information, refer to Space Replication. |
| SingleByteObjectSize | Defines the single byte objectsize for the cache server. |
| MultiByteObjectSize | Defines the multi byte objectsize for the cache server. |
| Limit | Defines the limit in which the memory is divided among all cache types, such as, Local, Distributed, and Near.<br>• If the Limit is specified, list size is ignored.<br>• If limit and list size are not specified for Local and Near caches, the list size is defaulted as `100`.<br>• It is always recommended to specify limit or list size for Near caches to avoid mismatch between the capacity in heap and in distributed storage. |
| ListSize | Defines the list size of the cache. Use the ListSize to specify the exact capacity. Similarly, specify the Limit if you want the cache to get memory as it grows.<br>• To specify the unlimited cache size for the Distributed cache type, remove the <Limit> attribute and specify ListSize = `-1`.<br>• For Near and Local cache types, the unlimited cache size is not supported. |

| Cache Attributes | Descriptions |
|---|---|
| EvictionPolicy | Defines eviction policy for a cache. The eviction polices are Least Recently Used (LRU) and None. By default, the eviction policy is set at `LRU`.<br>• If a capacity (number of objects in a cache) is specified, you must specify an eviction policy.<br>• For local caches, `eviction = LRU` is always applied and you cannot specify any other eviction policy.<br>• For distributed caches, `LRU` or `NONE` must be specified.<br>• If you want to keep the caches permanently, do not set `LRU`.<br>• If Limit is specified and EvictionPolicy is not specified, `LRU` is applied to those caches.<br>• If EvictionPolicy (LRU) is set, the least recently used record are deleted and new record are inserted.<br>• If EvictionPolicy (LRU) is not set, and the Capacity crosses its limit, the cache does not allow the insertion of a record and the new record insertion fails.<br>**Note**: It is recommended that unless instructed by Support, you should not change the default EvictionPolicy defined in the `CacheConfig.xml` file. |
| LockWaitTime | Defines the value while waiting for the lock to clear. An application can lock an entry so that the entry cannot be modified (but can still be read) until the lock is explicitly removed. By default, it is set to `-1`, which means the lock never waits. |
| LockExpirationTime | This property defines the value while lock for an entry expires. By default, it is set to `-1`, which means the lock never expires. |

## Cache Configuration Example

The sample shows an example of Cache attributes specified in the `CacheConfig.xml` file.

```
<CacheConfig>
    <ServerConfig>
                <CacheServerCount>1</CacheServerCount>
                <CharSet>singlebyte</CharSet>
                <Memory>512</Memory>
                <HeapStorage>128</HeapStorage>
                <OverHead>1.5</OverHead>
                <ReplicationCount>0</ReplicationCount>
        </ServerConfig>
            <CacheList>
                      <Cache>
                                <Name>WORKFLOWEXPR</Name>
                                <Type>local</Type>
                                <ReplicationCount></ReplicationCount>
                                <SingleByteObjectSize></SingleByteObjectSize>
                                <MultiByteObjectSize></MultiByteObjectSize>
                                <Limit></Limit>
                      </Cache>
                  <Cache>
                                <Name>RECORD</Name>
                                <Type>distributed</Type>
                                <ReplicationCount></ReplicationCount>
                                <SingleByteObjectSize>1320</SingleByteObjectSize>
                                <MultiByteObjectSize>1920</MultiByteObjectSize>
                                <Limit>24.0</Limit>
                      </Cache>
        </CacheList>
</CacheConfig>
```

You can switch the cache configurations for different environments, such as, large and development. For large number of repositories or large data, use the `CacheConfig.large.xml` file. For development environment, use `CacheConfig.dev.xml` file. The files are located in `$MQ_HOME`. As per your requirement, rename the file to CacheConfig.xml.

# CacheManager Utility

The CacheManger utility (`CacheManager.bat` or `CacheManager.sh`) is added in `$MQ_HOME\bin` directory. Using this utility, you can get the statistics of ActiveSpaces cache and member details.

The usage of the utility is as follows:

CacheManager [options]

Options:

`-?` : Print usage

`-connect` : Connects to the metaspace.

`-Listen` : Listen space for [Put/Take/Expire/Seed/Unseed] activity.

`-member` : Displays information about members.

`-space` : Displays space details.

`-search` : Searches the records in space.

`-asadmin` : Executes the exact `asadmin` commands.

## Example To listen space for Put Take Expire Seed Unseed activity

-s <string> Space name

-q <string> Filter condition (For example, listen only specified matching record operation)

Examples:

```
Start listening space: listen -s RECORD
Start listening space with filter: listen -s RECORD -q t_id>0
Stop listening space: stop listen <space_name>
Stop listening all spaces: stop listen all
```

## Example To display information about members

-n <string> Member name

Examples:

```
Display all member list: members
Display all members with full details: members detail
Display details of sepcified Member: member -n <member_name>
```

## Example To display space details

-s <string> Space name

Examples:

```
Display space details: space -s <space_name> Note : space_name is case senstive
Display all spaces: spaces
```

### Example To search for records in space

search -s <string> -q <string> Display all records containing the specified string in AS Format

Examples:

```
Display all records satisfies specified condition: search -s RECORD -q z_id like
".*RECORD__16082.*"
```

### Example To execute exact asadmin commands

Examples:

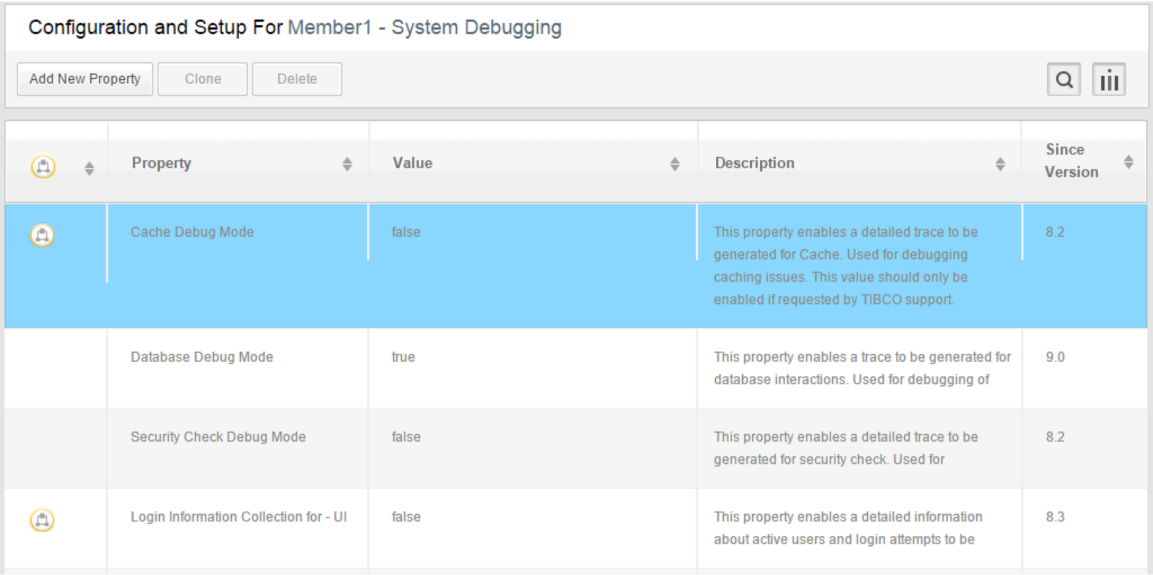
```
Display all spaces: asadmin show spaces
Display all spaces: asadmin show space "<space_name>"
```

## Tracing and Controlling the Cache

When the application is configured to run in the debug mode, the cache subsystem generates a large number of debug messages. This causes the logs to fill up quickly. Suppress the debug messages generated from the cache subsystem.

#### Procedure

1. Log in to Configurator.
2. Select **Node ID** > **System Debugging** .
3. Search for the `Cache Debug Mode` property to enable or disable the cache tracing.



Configuration and Setup For Member1 - System Debugging

| | Property | Value | Description | Since Version |
|---|---|---|---|---|
| | Cache Debug Mode | false | This property enables a detailed trace to be generated for Cache. Used for debugging caching issues. This value should only be enabled if requested by TIBCO support. | 8.2 |
| | Database Debug Mode | true | This property enables a trace to be generated for database interactions. Used for debugging of | 9.0 |
| | Security Check Debug Mode | false | This property enables a detailed trace to be generated for security check. Used for | 8.2 |
| | Login Information Collection for - UI | false | This property enables a detailed information about active users and login attempts to be | 8.3 |

By default, the value is `false`, which indicates that debug messages are not generated. To enable the cache tracing, specify the `true` value.

## Memory Calculation for Cache

When considering memory for cache consider the a couple scenarios.

- Memory available for Local and Near caches = heap storage Assume, it is A.

- Memory available for Near and Distributed caches = Memory - heap storage Assume, it is B.

  In these scenarios, the memory assigned to any cache is calculated as follows:

  – If the Limit is specified, the ListSize is ignored. Limit is the whole number and a percent. The minimum value is `1` and the maximum value is `99`. However, a fraction is also supported with a single digit. The minium value is `0.1` and the maximum value is `99.9`.

    [(Limit/100) * B]/ObjectSize = Capacity

    In this case, the capacity is a number of objects available in a cache.

  – If the Limit and ListSize are not specified, the memory for Local cache is set to default 100 value. For Distributed cache, no default value is set.

  1. Capacity = List Size, if specified.
  2. For the Distributed cache, if ListSize is not specified,

     Count all distributed caches, which do not have ListSize or Limit. For example, N

     Assign memory to all those caches, which have ListSize or Limit. For example, C

     ```
     Remaining memory = B - C= D
     ```

     Available memory for one cache = E = D/N

     Capacity = E/ObjectSize.

- Memory consumed by each cache = capacity * ObjectSize.

If the ListSize is specified as `-1` and the Limit is not specified, this indicates the unlimited cache. Such caches are excluded from the memory computation and they have no limit.

If the object sizes are not correct, the actual memory allocation may be different from the total assigned memory allocation.

It is recommended to:

- Allocate 20% more than the actual memory.

  – If the allocated heap size within the range tolerance of 5MB, a warning is displayed. However, if the tolerance exceeds, an error is displayed.

  – If the distributed cache size exceeds the tolerance of 20MB, a warning is displayed. However, if the tolerance exceeds, an error is displayed.

    To rectify the error that occurs when the allocated memory exceeds the specified memory, fix the allocation. If it does not resolve the error, specify the `true` value for Cache Memory Check property in Configurator. By default, the value is `false`.

- Specify the ratio between ProductKey to Record as **1: 3.3** and ProductKey to RecordMaxModVersion as **1:2.3.**

- For large installations, it is recommended that RECORD, RECORDMAXMODVERSION, and PRODUCTKEY be set as unlimited.

## View Memory Allocation

To view the memory allocation, verify the startup log in the *elink.log* file.

- Memory Allocation Count

The following log indicates the Server configuration. You can identify the allocated memory count.

```
2013-01-24 10:37:34,839 [MSC service thread 1-10] DEBUG
com.tibco.mdm.infrastructure.cache.as.CacheConfigParser - Server configuration:
{Name=Dev configuration, CharSet=true, Memory=512, ReplicationCount=0,
HeapStorage=50, OverHead=1.3,
```

- Total Memory Allocation

The following example indicates whether or not all of the heap storage memory used.

```
2013-01-24 10:37:34,891 [MSC service thread 1-10] INFO
com.tibco.mdm.infrastructure.cache.as.CacheSizeUtility - Distributed and near
caches: Available Memory (MB) = 462.0 Allocated Memory (MB) =462.8532301477959
```

To know whether allocated memory is sufficient or not, check the hit ratios in the Cache MBeans using JConsole. For more information on Cache MBeans, refer to Appendix B TIBCO MDM Management Using JMX in *TIBCO MDM System Administration guide*.

### Object Size Calculation for Cache

Calculate the object size for cache with this method.

The ObjectSize is calculated as follows:

- If <CharSet> is singlebyte, check if the <SingleByteObjectSize> is specified. If not specified, default 100 value is considered.
- If <CharSet> is multibyte, check if <MultiByteObjectSize> is specified. If not specified, default 200 value is considered.
- For distributed caches, ObjectSize = ObjectSize * overhead factor

  A sum of all percent limits for Distributed and Near caches must be 100 or a sum of all percent limits for Local and Near caches must be 100 or less.

## Cache Computation

As the application ages and whenever more data is added, you may observe that the cache is incorrectly sized. Memory calculation of cache is also quite complex in TIBCO MDM. Using the auto compute option of the cache, auto cache calculation is suppressed depending on the capacities of caches.

By default, TIBCO MDM adds different types of overhead and factors in capacity calculation, however, the approximate capacity can be calculated using the following generic formula:

```
Capacity = Total number of objects exist in the database * number of entries
maintained in cache per record
```

For example, `ENTERPRISE` cache object, assume five enterprises are available in the database and tibco MDM maintains two entries in cache per enterprise, the capacity for the `ENTERPRISE` cache object is `5*2 = 10`.

By default, the **`Cache Calculation Suppression`** ( **`com.tibco.mdm.cache.autocompute.disable`**) property is set to `false`. If you change the value to `true`, the cache calculation is suppressed.

| Cache Name | Description |
|---|---|
| Enterprise | Counts the number of enterprises, and derives capacity of the following caches:<br>• `ENTERPRISE`, `ORGANIZATION`, `HTTP`, `CONFIGDEFINITION`, `CONFIGDEFINITIONLIST`, `RULEENGINE`, and `RULEENGINEATTRIBUTEINFO` |

| Cache Name | Description |
|---|---|
| Member | Counts the number of users and derives the capacity of the following caches:<br>• MEMBER, MEMBERORG, WORKITEM, WORKITEMLOCK, WORKITEMPREFERENCES, and WORKITEMSUMMARY<br>• CONFIGURATIONDOCUMENT, MASSUPDATE, MATCHEROUTPUT, RULEBASERESULTS, and PRODUCT<br>• PRODUCT, RECORDBUNDLE, SEARCHCRITERION, SECURITYPERMISSIONS, andLOCALSUBCATALOG |
| Repository | Counts the number of repositories and derives the capacity of the following caches:<br>• CATALOG, SECUREDATTRIBUTEGROUP, ASSOCSCHEMES, CATALOGFORMATS, CATALOGEDITIONVALUE, and CATALOGEDITION<br>• CATALOGINPUTMAP,OUTPUTMAPLIST, RELATEDMAPS, CLASSIFICATIONCODE, and CLASSIFICATIONCODEATTRIBUTE<br>• CATALOGDECLARATION, RULEBASE, RELATIONSHIPMETADATA, and PRELOAD_STATUS |
| Event | Assumes the minimum events to be stored to 2000 and derives the capacity of the following caches:<br>• EVENT, EVENTDETAIL, PROCESS, MLXMLDOC, PROCESSSTATE, and PROCESSLOG<br>• RECORDCOLLECTION, RECORDKEYLIST, and RECORDITEM |
| PoolSize | Inquires the pool size of the workflow receiver and configures the following caches:<br>• WORKITEMDOC and COMPUTEDSUMMARY<br>• LOCAL except LOCALSUBCATALOG |

After the cache size is calculated and when the caches are being created, the cache size configured through the CacheConfig.xml file is compared, and the higher value is used. Thus, the total memory allocation to the cache can be higher than what is configured through CacheConfig.xml. To avoid startup issues, when cache capacities are auto computed, the excess memory check is suppressed and the server allocates more memory than configured. To verify cache computation, the following message is generated in debug log:

```
“Autocompute enabled, excess memory allocation allowed.”
```

RECORD cache are not included in the auto computation.

## Repository Spaces for Sharing Cached Data

Use *Repository Spaces* to share the cached data with external applications. TIBCO MDM caches the record data in the distributed cache.

Using Repository Spaces, you can:

- Cache records as a table and not as a BLOB.
- Cache relationship attribute records.

- Create a repository space for each repository. For more information, refer to Repository Spaces - An Overview.
- Maintain data in the cache even if the record is updated.

## Repository Spaces - An Overview

The repository space is the information of a repository stored in ActiveSpaces. A repository space is created when TIBCO MDM starts. However, the space is created after the repository is confirmed.

All existing repositories are searched, and the spaces are created. The repository space is optimized for a cluster so that every member of the cluster does not attempt to create the space.

The following list describes the relationship between the repository and its space:

- If you delete a repository, the space is not deleted. It remains as is until the TIBCO MDM server restarts.
- If you add an attribute in the repository, a new column is created in the space according to the data type of the attribute and database column name. The default value of the column is null. The records that were created before adding the attribute contain null value in the column.
- If you delete an attribute from the repository, the attribute is not deleted from the space. It maintains null value for the incoming records. However, after restarting TIBCO MDM, the deleted attribute is not created in the space.
- As each record is stored in its own repository space, object type= RECORD_*RepositoryID* is mapped to the repository space. The mapping is performed as RECORD_*Repository Table name*.

## Repository Spaces Configuration

By default, `false` is specified for the Enable repository spaces property in Configurator (**Initial Config** > **Optimization**). Specifying `false` does not create the repository spaces. Select `true` to enable caching of the records in repository spaces.

If you enable the repository spaces, the list size of the cache might not be sufficient and an error is displayed at the server start-up. To avoid the error, specify the ListSize = `-1` in the `CacheConfig.xml` file.

When you perform record operations in TIBCO MDM, specify `true` for the Cache Debug Mode property in Configurator. You can confirm whether the data is retrieved through the cache or through the database, and verify the logs. For more information, refer to the Tracing and Controlling the Cache section in *TIBCO MDM System Administration*.

## Organization of Repository Space in ActiveSpaces

The repository space is organized in ActiveSpaces in different ways.

The repository space is organized in the following ways:

- The fields of the space are named using the database column name assigned to each attribute. Therefore, the space appears similar to the database table.
- All the predefined attributes are prefixed with Z_. The predefined attributes are displayed in ActiveSpaces as follows:

***Mapping of the Column, Space Field, and Data Type***

| Column Name | Space Field (Predefined Attributes) | Data Type, if different |
|---|---|---|
| `PrincipalKey. MODVERSION, MCT.CMODVERSION` | `Z_MODVERSION` | |

| Column Name | Space Field (Predefined Attributes) | Data Type, if different |
|---|---|---|
| `PrincipalKey.CATALOGID` | `Z_CATALOGID` | |
| `PrincipalKey.CATALOGVERSIONNUMBER` | `Z_CATALOG_VERSION` | |
| `PrincipalKey.MODDATE` | `Z_MODDATE` | `LONG` |
| `PrincipalKey.ACTIVE` | `Z_ACTIVE` | |
| `PrincipalKey.MODMEBERID` | `Z_MODMEBERID` | |
| `PrincipalKey.CREATIONDATE` | `Z_CREATIONDATE` | `LONG` |
| `PrincipalKey.LASTIMPORTTIME` | `Z_LASTIMPORTTIME` | |
| `PrincipalKey.CHECKSUM` | `Z_CHECKSUM` | |
| `PrincipalKey.LASTCONFRIMEDVERSION` | `Z_LASTCONFRIMEDVERSION` | |
| `PrincipalKey.OWNERID` | `Z_OWNERID` | |
| `PrincipalKey.PRODUCTKEYID` | `Z_PRODUCTKEYID` | |
| `PrincipalKey.STATE` | `Z_STATE` | |

- Due to the inherent limitation in ActiveSpaces, some data types cannot be mapped as is. The following table describes the mapping of the TIBCO MDM data types to the ActiveSpaces types:

*Mapping of the MDM Data Types to ActiveSpaces Types*

| TIBCO MDM Data Types | ActiveSpaces Types |
|---|---|
| String | String |
| Boolean | Boolean |
| Date | Long |
| Long (IDs) | Long |
| Integer | Integer |
| Decimal, Custom Decimal, Amount, Number | Float |
| Timestamp | Long |
| FILE | String |

## Viewing Spaces in ActiveSpaces Monitoring and Management

You can view various aspects of repository spaces using ActiveSpaces Monitoring and Management (ASMM).

- Repository Spaces
- Predefined and Custom Attributes
- Relationship Records
- Multivalue and Category Specific Attributes

For information on logging in to ActiveSpaces Monitoring and Management, refer to *TIBCO ActiveSpaces Administration*.

### Procedure

1. Log on to ASMM.
2. View Repository Spaces
   a) Click the RECORD_*RepositoryTableName* repository space in the **Metaspace Navigator** column. For example, if the repository table name in TIBCO MDM is MCT_36017 , the repository space name is displayed as RECORD_MCT_36017.

      The repository space name along with the space information is displayed on the right panel.
3. Viewing Predefined and Custom Attributes
   a) Click the RECORD_RepositoryID repository space in the **Metaspace Navigator** column. For example, RECORD_MCT_36017. The repository space name along with the space information is displayed on the right panel.
   b) Click the **Schema** tab. The Schema tab displays:

   - All attributes of a repository in the Name column.
   - All predefined attributes are prefixed with Z_. For example, Z_ PRODUCTKEYID
   - All custom attributes are defined by their Column Name, that is, C*AttributeName*. For example, CBANKNAME.
   - The dates are mapped to the LONG data type. For example, CDATEOFOPEN attribute.
   - The Amount, Decimal, and Custom Decimal attributes are mapped to the FLOAT data type attribute. For example, CBALANCE attribute.
   - The Timestamp and EFFECTIVEDATE attributes are mapped to the LONG data type.
   - The File type attribute is mapped to the STRING data type. For example, CPICSOFACCOUNTHOLDER. The filename is displayed when you view the records in cache.
4. Viewing Relationship Records

   All relationships which contain attributes are stored in its own space with the name RECORD_*Relationship Table Name*. You can identify the Relationship Table Name in TIBCO MDM on the View Repository page. Therefore, in this case the space name for AddressToAccount relationship is RECORD_*RCT_36072*.

   a) Click the RECORD_RCT_36072 repository space in the **Metaspace Navigator** column. The repository space name along with the space information is displayed on the right panel.
   b) Click the **Schema** tab. The **Schema** tab displays the following information:

   - All predefined attributes prefixed with Z_. For example, z_id, Z_ TYPE, and z_CATALOGID.
   - Relationship attributes displayed by their database column names mentioned in TIBCO MDM with the appropriate data type. For example, CHEADOFFICE and CBRANCHNAME.

## Multivalue and Category Specific Attributes

For each multi-value and category specific attributes, individual fields are created in repository spaces with their database column name.

- The multi-value attribute field is created as the string data type and its values are stored as a delimiter separated string.
- The category specific attribute field is created with the same type that is stored in the database.

The following configuration properties for multi-value and category specific attributes are created in Configurator:

*Configuration Properties for Multi-value and Category Specific Attributes*

| Property Name | propname | Description | Value |
|---|---|---|---|
| `Average number of values in a single multivalue attribute` | `com.tibco.cim.cache.multivalue.avaerage.values.perattribute` | Indicates the average number of values to be specified in a single multi-value attribute when repository space is enabled. | Any valid integer.<br>The default value is `2`. |
| `category specific attribute size (bytes)` | `com.tibco.cim.cache.categoryspecific.attribute.size` | Refers to the category specific attribute size to be used for repository space size calculation. The size is specified in bytes. | Any valid integer.<br>The default value is `10`. |
| `ActiveSpace multivalue qualifier` | `com.tibco.cim.cache.multivalue.qualifier` | Refers to the multi-value qualifier specified in multi-value attribute values that is used for cache. | Any valid qualifier.<br>The default is `&quot;`. It signifies the double quotation marks. |

The existing Delimiter used while reading multi value data property is used while reading multi-values in ActiveSpaces. The default value is !#.

## Object Size Calculation for Repository Spaces

An object refers to the row of a record included in the space. You can calculate the object size of the repository space. Each repository space has a different object size based on its attributes. Therefore, the object size calculation for each repository space is different.

For example, to calculate the object size of 10,000 records included in the Customer repository. The object size of a record is calculated as follows:

```
Object Size = (Sum of all catalog attributes (excluding multivalue and category
specific attribute) data types lengths * (2 Or 3)) +  (All multivalue attributes
size) + (All category specific attributes size)
```

To calculate the object size for a repository space, use the following formulas:

*Object Size Calculation for Repository Spaces*

| Formula | Usage |
|---|---|
| `Sum of all repository attributes (excluding multivalue and category specific attribute) data types lengths = (attribute1 data type Length *2) + (attribute 2 data type length*2) + ……… (attribute n length *2)]` | Retrieves single byte object size. |
| Sum of repository attributes' data types lenght = [(attribute1 data type Length *3) + (attribute 2 data type length*3) + (attribute n length *3) | Retrieves multi-byte object size. |

If the attribute data type is String, Attribute length is multiplied by 2 or 3 bytes. Else, the attribute size is considered as its data type size. For example, for the Integer data type it is considered as 4 bytes.

To calculate the actual object size, 1/3rd of the calculated object size is considered and ROWOVERHEAD is added.

```
Actual object size of a repository = Object Size/3 + ROWOVERHEAD
```

The value of ROWOVERHEAD is configured in the <ServerConfig> section of the `CacheConfig.xml` file. It is the sum of System Attributes Overhead, ActiveSpaces overhead, and Reference keys overhead. For more information, refer to RowOverHead and MinimumSeederRequiredForPreload Attributes Configuration.

## ObjectSize Calculation for Multi-value and Category Specific Attributes

Use these queries to calculate the ObjectSize of the multi-value and category specific attributes.

*Object Size Calculation for Multi-value and Category Specific Attributes*

| Query | Usage |
|---|---|
| `The string length of maximum value of the multi-value database column type * Single Byte or Multi Byte object size * average number of entries in a multi-value attributes`<br>In this way,<br>• If Single byte: Total size of multi-value attribute = length of data type * 2 * average number of values<br>• If Multibyte byte: Total size of multi-value attribute = length of data type * 3 * average number of values | Retrieves multi-value attribute size. For example,<br>• If a column contains the Integer data type, its maximum value is 2,147,483,647. The integers are converted into string. Therefore, these are considered as 10 characters, and eventually its size is considered as 10.<br>• If a column contains the String data type, its size is considered as Length of the String defined in database. |
| For each category specific attribute, the size that is configured in Configurtor is used for calculation. For information on the property, refer to Configuration Properties for Multi-value and Category Specific Attributes. | Retrieves category specific attribute size. |
| `Total ObjectSize = Object Size calculated from Database for repository space + the size of all multi-value attributes columns + category specific attributes' size` | |

## RowOverHead and MinimumSeederRequiredForPreload Attributes Configuration

The RowOverHead and MinimumSeederRequiredForPreload attributes are added to the <ServerConfig> section of the `CacheConfig.xml` file.

- RowOverHead: This is an additional overhead. The RowOverHead attribute is added to calculate the actual object size of a repository. Each repository space record contains the RowOverHead. It is the sum of the system attributes overhead, ActiveSpaces overhead, and all reference keys overhead.
- MinimumSeederRequiredForPreload: A number of seeders are required to initiate the preload. When preload initiates, each node verifies the number of seeders who have joined the RECORD space. If the seeders count is equal to or greater than the value specified for MinimumSeederRequiredForPreload, the preload represents the distributed cache objects. Else, the preload represents only the local and near cache objects.

### Example

```
<ServerConfig>
   <Name>Standard configuration</Name>
   <CacheServerCount>1</CacheServerCount>
   <CharSet>singlebyte</CharSet>
   <Memory>2048</Memory>
   <HeapStorage>300</HeapStorage>
   <OverHead>1.3</OverHead>
   <ReplicationCount>0</ReplicationCount>
   <RowOverHead>2170</RowOverHead>
   <MinimumSeederRequiredForPreload>1</MinimumSeederRequiredForPreload>
</ServerConfig>
```

## Configuring Custom Repository

You can manually configure a repository in the `CacheConfig.xml` file. The repository space is created for the configured repository, and the configuration parameters are retrieved from the `CacheConfig.xml` file instead from the RECORD cache.

For example, to configure the customer repository in the `CacheConfig.xml` file:

### Procedure

1. Locate the table name for the Customer repository in the database.
2. Add the RECORD prefix to the repository table name and configure the repository in the `CacheConfig.xml` file as follows:

```
<Cache>
   <Name>RECORD_$CUSTOMER_TABLE_NAME</Name> <!-Mandatory -- >
   <Description>This is a custom defined repository space</Description> <!-
Optional -- >
   <Type>distributed</Type> <!-Mandatory -- >
   <SingleByteObjectSize></SingleByteObjectSize> <!-Mandatory -- >
   <MultiByteObjectSize></MultiByteObjectSize> <!-Mandatory -- >
      <ReplicationCount></ReplicationCount> <!-Optional -- >
      <Limit></Limit>   <!-Optional -- >
      <ExpirationTime></ExpirationTime> <!-Optional -- >
    <EvictionPolicy></EvictionPolicy> <!-Optional -- >
    <LockWaitTime></LockWaitTime> <!-Optional -- >
    <LockExpirationTime></LockExpirationTime> <!-Optional -- >
</Cache>
```

3. Save the `CacheConfig.xml` file.

### Validation Errors for Repository Spaces

When you deploy repository metadata in TIBCO MDM Studio, the repository space is validated for its table name. The validation error messages are displayed in the Repository Model Validation dialog box.

- If a repository space exists with the specified table name, the following validation error is displayed:

  ```
  CACHE-7561: Repository space '$REPOSITORY_SPACE_NAME' already exists. Specify
  the unique name.
  ```

- If the table name contains special characters or an invalid name, the following validation error is displayed:

  CACHE-7562: Invalid repository space name '$REPOSITORY_SPACE_NAME'.

## Space Locking

To acquire lock in ActiveSpaces, a distributed object MDMLOCKSPACE is defined in Configurator. When TIBCO MDM server starts, the MDMLOCKSPACE is created in ActiveSpaces.

For multiple seeder cache node, it is recommended to replicate MDMCLOCKSPACE. Refer to the following sample:

```
<Cache>
<Name>MDMLOCKSPACE</Name>
<Description>This space maintains MDM locks</Description>
<Type>distributed</Type>
<ReplicationCount>0</ReplicationCount>
<SingleByteObjectSize>548</SingleByteObjectSize>
<MultiByteObjectSize>818</MultiByteObjectSize>
</Cache>
```

The MDMLOCKSPACE contains following fields:

***MDMLOCKSPACE Fields***

| Fields | ActiveSpaces Data Type | Description |
|---|---|---|
| `z_id` | String | Refers to the cache key. |
| `THREADID` | Long | Refers to the thread ID that acquires the lock. |
| `THREADNAME` | String | Refers to the thread name that acquires the lock. |
| `DATETIME` | Datetime | Indicates the date and time when an entry is locked. You can use the date and time to identify the age of lock. The date and time is stored in the GMT format. |
| `NODEID` | String | Indicates the node ID. Using the combination of NODEID and THREADID, you can identify which node and thread acquired or released the lock. |

The MDMLOCKSPACE is an important space that contains the ActiveSpaces lock information. Therefore, it is recommended that you must replicate this space. For MDMLOCKSPACE, if you do not specify <ReplicationCount> as greater than zero, the following warning message is displayed in elink.log:

```
***** Replication count is not set for MDMLOCKSPACE ********
```

## Space Replication

To provide fault-tolerance and prevent loss of cached data in any of the space (if one of the seeders of a space suddenly disappears from the Metaspace), specify a degree of replication for a space.

Replication in ActiveSpaces is performed in a distributed active-active manner. Seeders seed and replicate some data assigned to other seeders. The replication itself is distributed, rather than keeping a designated backup for each seeder. This backup replicates all of the data that the seeder seeds, and the data that it seeds are replicated by all of the other seeders.

The <ReplicationCount> attribute is available for each space in the `CacheConfig.xml` file.

The loss of data for the following caches results in application failure and reduced performance. Therefore, replicate these caches:

- COUNTERS
- ACTIVITYRECORDCOUNTER
- FAILOVERMARKER
- FILELOCK
- SPACE_LOCKS
- MDMLOCKSPACE

## Perspective Space

To support the perspective cache, the `PERSPECTIVE` cache object is defined in the `CacheConfig.xml` file.

Refer to the following sample:

```
<Cache>
<Name>PERSPECTIVE</Name>
<Description>This space maintains Perspective data</Description>
<Type>distributed</Type>
<ReplicationCount>0</ReplicationCount>
<SingleByteObjectSize>240</SingleByteObjectSize>
<MultiByteObjectSize>320</MultiByteObjectSize>
<ListSize>50</ListSize>
</Cache>
```

By default, perspective uses the hibernate technology to interact with the database. However, if you want to use JDBC related libraries, change the value to `false` of the **`com.tibco.mdm.perspective.hibernate.enable`** property in Configurator.

### Cache Operation on Perspective

- Whenever a perspective is deployed, it is added or updated in cache.
- If a perspective is deleted, its entry is removed from cache except the mod version entry.
- If a repository is deleted, perspective is removed from cache.
- If a perspetive is modified, the perspective cached is also updated.

Whenever a perspective is created, the following entries are made in cache:

- PERSPECTIVE__*PERSPECTIVEID*
- PERSPECTIVE__*PERSPECTIVEID__MODVERSION*
- PERSPECTIVE__*CATALOGID__NAME*
- PERSPECTIVE__ REL__ *PERSPECTIVEID__MODVERSION*
- PERSPECTIVE__ATTR__*PERSPECTIVEID__MODVERSION__CATALOGID__RELATIONSHIPDEFINITIONID* __DIRECTION

Only when relationships and attributes are accessed, they are cached in ActiveSpaces.

## Perspective Cache Keys

Perspective is stored in cache with the following keys:

| Cache Key | Description |
|---|---|
| All cache key names start with a prefix `PERSPECTIVE__` | |
| *PERSPECTIVE_ID* | Stores perspective information by perspective ID. It contains only perspective information, no information about relationships and attributes. |
| *CATALOG_ID__PERSPECTIVE_ID* | Stores perspective information by combination of catalog ID and perspective name. It contains only perspective information, no information about relationships and attributes. |
| *CATALOG_ID* | All perspectives which are associated with this catalog. It contains only perspective information, no information about relationships and attributes. |
| *PERSPECTIVE_ID__MODVERSION* | Stores perspective information for specified modversion. No information about relationships and attributes. |
| `REL__`*`PERSPECTIVE_ID__MODVERSION`* | Stores all relationships, which are defined in perspective with the provided modversion. |
| `ATTR__`*`PERSPECTIVE_ID__MODVERSION__CATALOG_ID__RELATIONSHIPDEFINITIONID__DIRECTION`* | Stores all attributes, which are defined in perspective with the provided modversion for the specified repository and selected relationship definition. |

# Bundle Caching

In large bundle scenarios, that is, in cases more than thousands of records per bundle, the loading of a bundle can take a long time and puts a large stress on the database. The bundle is loaded in steps for each parent and immediate child in the overall bundle, which leads to many queries against database. By using bundle caching, the full bundles are loaded into the cache and can be retrieved from the cache instead of the database for the next use of the bundle.

### How Bundle Caching Works

A bundle is the connected set of records based on a root record, where each of the dependent nodes is discovered in depth, the first traversal of the record space. The root record defines the bundle identity. Along with the root record, the following parameters are part of the cached bundle:

- Bundle depth
- Relationship definitions to be traversed
- Relationship definition depth

Bundle Caching is currently only implemented in a limited fashion. You can use the bundle caching for the following record operations:

- Modify record through web service: when you modify a record in the bundle, the bundle cache is searched for all bundles containing the updated record, and each of these bundles from the cache is updated with the new record.
- Validate record through web service
- Query record through web service: when you query a record through web service, the bundle is added to the cache.
- If you Query record with perspective as the context parameter, perspective related bundle is added to the Bundle Cache.

All these use cases require repeated usage of the bundle to see the performance gains.

## Bundle Cache Space

To load the golden record bundles as cache in ActiveSpaces, a distributed object `BUNDLE` is defined in the `CacheConfig.xml` file.

Refer to the following sample:

```
<Cache>
        <Name>BUNDLE</Name>
        <Description>This space contains global record bundles as cache</
Description>
        <Type>distributed</Type>
        <ReplicationCount>0</ReplicationCount>
        <SingleByteObjectSize>3000</SingleByteObjectSize>
        <MultiByteObjectSize>4500</MultiByteObjectSize>
        <ListSize>500</ListSize>
</Cache>
```

The `BUNDLE` space contains the following fields:

| Field Name | ActiveSpaces Data Type | Description |
|---|---|---|
| `z_id` | String | Refers to the bundle ID. This is a primary key. |
| `DATE` | Datetime | Indicates the bundle creation date, using which you can evict bundles out of the cache based on the staleness |
| `RKEYLIST` | String | Refers to the record key list that contains the record IDs concatenated with a separator ‘__’ |
| `BUNDLESTRUCT` | String | Contains hints about the graph structure of the bundle |
| `RELATIONSHIPDEF` | String | Encodes a list of relationships. The list is an encoded form of the IDs in the **relationshipMetaInfo** field in the Recordbundle cache. |
| `ALLRELATIONS` | Blob | Serializes the map object (relationships per relationship definition) so that the **allRelationship** field inside the Recordbundle cache can be reestablished upon record bundle access from cache. |
| `rkeyidx` | String | Indicates an index on the record key list for a quick search on the field. |

## Configuration Properties for Bundle Caching

*Configuration Property for Bundle Caching*

The following table lists the property of Bundle Caching specified in Configurator:

| Property Name | Location | Value | Description |
|---|---|---|---|
| Enable Record Bundle Caching (`com.tibco.mdm.bundlecache`) | Cluster Level (InitialConfig > Optimization) | `True` or `False`. The default value is `False`. | Enables bundle caching. |

# Apache Ignite

Apache Ignite is a high-performance, integrated and distributed in-memory platform for computing and transacting on large-scale data sets in real-time.

The performance of TIBCO MDM highly depends on the distributed cache. Apache Ignite meets various cache requirements of TIBCO MDM and can be used as the caching option.

Apache Ignite as a collection of independent, well-integrated, in-memory components improves the performance and scalability of an application. TIBCO MDM uses core caching data-grid module of Apache Ignite with advanced indexing and SQL capability.

For more information on Apache Ignite, refer to Apache Ignite documentation.

## Enable Apache Ignite for TIBCO MDM

Enable the usage of Apache Ignite as distributed cache for TIBCO MDM using the following steps:

- Select `Ignite` for the `Cache Type` (`com.tibco.cim.cache.type`) property in Configurator. Other optional configuration properties are listed in the Configuration properties of Apache Ignite section.
- Specify JVM parameters listed in the JVM arguments for Apache Ignite section.
- Use `IgniteMember.xml` to configure the cache server topology. The default configuration restricts the single node topology. For more details on configuring the multinode topology, refer to `IgniteMember.xml` based cluster topology configuration

### Configuration Properties of Apache Ignite

*Configuration Properties for Apache Ignite*

The following table lists the properties of Apache Ignite specified in Configurator:

| Property Name | Location | Value | Description |
|---|---|---|---|
| Cache Type (`com.tibco.cim.cache.type`) | Cluster level (InitialConfig > Optimization) | The available options are `AS`, `Ignite`, and `Custom`. The default value is `AS`, that is, ActiveSpaces. | Type of cache used in TIBCO MDM. To use Apache Ignite as cache for TIBCO MDM, select `Ignite` for this property. |
| Ignite members configuration (`com.tibco.mdm.ignite.config.filename`) | Cluster level (InitialConfig > Optimization) | The default value is `IgniteMember.xml`. | Apache Ignite cache cluster configuration file name, relative to `$MQ_CONFIG` |
| Ignite Log Logging Level Default (`log4j.logger.org.apache.ignite`) | Node level (Member1 > Logging > Default Ignite Log) | The default value is `INFO,ignitelogdefault`. | Logging level for the ignite log |
| Ignite Log File Name Default (`log4j.appender.ignitelogdefault.File`) | Node level (Member1 > Logging > Default Ignite Log) | The default value is `${MQ_LOG}/ignite.log`. | Path and name of the log file |

| Property Name | Location | Value | Description |
|---|---|---|---|
| Ignite Log Maximum File Size Default (log4j.appender.ignitelogdefault.MaxFileSize) | Node level (Member1 > Logging > Default Ignite Log) | The default value is 5MB. | Maximum size of the ignite log file |
| Ignite Log File Backup Count Default (log4j.appender.ignitelogdefault.MaxBackupIndex) | Node level (Member1 > Logging > Default Ignite Log) | The default value is 40. | Number of the ignite log backup files |
| Ignite Log Layout Pattern Default (log4j.appender.ignitelogdefault.layout.ConversionPattern) | Node level (Member1 > Logging > Default Ignite Log) | The default value is %d,[%t],%m%n. | Pattern of an entry in the ignite log file |
| Ignite Log Appender (log4j.appender.ignitelogdefault) | Node level (Member1 > Logging > Default Ignite Log) | The default value is org.apache.log4j.RollingFileAppender. | Appender used for the ignite log |
| Ignite Log Entry Layout (log4j.appender.ignitelogdefault.layout) | Node level (Member1 > Logging > Default Ignite Log) | The default value is org.apache.log4j.PatternLayout. | Layout of the ignite log entries |
| Ignite Log Appender Encoding (log4j.appender.ignitelogdefault.encoding) | Node level (Member1 > Logging > Default Ignite Log) | The default value is UTF-8. | Encoding used for the appender of the ignite log |
| Ignite Log Logging Level Threshold Default (log4j.appender.ignitelogdefault.Threshold) | Node level (Member1 > Logging > Default Ignite Log) | The valid values are FATAL, ERROR, WARN, INFO, and DEBUG. The default value is INFO. | Logging level threshold for ignite logging |

## JVM Arguments for Apache Ignite

Specify the JVM parameters of the application server that is used to deploy TIBCO MDM.

| JVM Arguments | Value | Description |
|---|---|---|
| `IGNITE_ATOMIC_CACHE_DELETE_HISTORY_SIZE` | `100` | Apache Ignite per cache delete history queue size. This queue handles the case where the same key is being inserted and deleted at the same time. For TIBCO MDM, 100 size is sufficient. If you do not set this parameter,the `Out-Of-Memory` error message is shown at the server startup. |
| `IGNITE_UPDATE_NOTIFIER` | The valid values are `true` or `false`. The default value is `true`. | When Apache Ignite is initialized, the new version check is automatically done, `System.out`. You can disable this version notification by setting the `IGNITE_UPDATE_NOTIFIER` property to `false`. |

## Configuring Cache and Cluster Topology for Apache Ignite

By default, the cluster topology configuration is specified in the `IgniteMember.xml` file located at `$MQ_HOME/config`.

- Cache configuration: The cache configuration is defined in the `CacheConfig.xml` file. By default, TIBCO MDM configures all caches to `ATOMIC`, `OFF-HEAP`, and `SWAP disabled`.
- Cluster topology configuration: TIBCO MDM supports the following two cluster topologies. You can configure both of them with Apache Ignite:
  - Peer-to-Peer
  - Centralized Cache Server

By default, `IgniteMember.xml` configuration restricts a single node cache topology by using the `localHost=127.0.0.1` property. To configure multinode distributed cache, remove the `localHost` property.

### Procedure

1. Navigate to the `$MQ_HOME/config` folder.
2. Open the `IgniteMember.xml` file.
   The following table explains various properties defined in the `IgniteMember.xml` file.

| Property Name | Description | Default Value |
|---|---|---|
| **Cache Configuration** | | |

| Property Name | Description | Default Value |
|---|---|---|
| `clientMode` | Defines the role of a node. For centralized cache server architecture, Apache Ignite can be started as a standalone process and TIBCO MDM connects to Apache Ignite cluster as a client node. To run TIBCO MDM in client mode, set `clientMode` to `true`. Client node cannot hold data in the caches. | `false` |
| `peerClassLoadingEnabled` | If `peerClassLoadingEnabled` is set to `true`, you do not have to manually deploy your Java code on each node in the grid and re-deploy it each time it changes. For performance reasons, the property is set to `false`. | `false` |
| `gridName` | Defines a name to Apache Ignite instance. It helps when multiple instances needed within same JVM. | MDMNODE |
| `localHost` | Gets system-wide local address or host for all components of Apache Ignite to bind to. If defined, it overrides all default local bind settings within Apache Ignite or any of its SPIs. If null, then Apache Ignite tries to use the local wildcard address. That means all services are available on all network interfaces of the host machine.<br>For cluster setup, you need to remove this property. | `127.0.0.1` |
| `metricsLogFrequency` | Metrics print frequency in Millisecond. The `zero` value indicates that no print is required. If the value is greater than zero and the log is not quiet, then statistics are printed with INFO level once a period. | `60000` (every minute) |

| Property Name | Description | Default Value |
|---|---|---|
| **Cluster Topology Configuration** | | |
| `ipFinder` | IP finder that shares information about the IP addresses of nodes. Nodes can discover each other by using `DiscoverySpi`. Apache Ignite provides `TcpDiscoverySpi` as a default implementation of `DiscoverySpi` that uses TCP/IP for node discovery. You can configure `DiscoverySpi` for multicast and Static IP based node discovery. | The default value is `TcpDiscoveryMulticastIpFinder`.<br>You can specify the `TcpDiscoveryVmIpFinder` implementation. |
| `localAddress` | Sets the local host IP address that `DiscoverySpi` uses | If not provided, by default a first found non-loopback address is used. If there is no non-loopback address available, then `java.net.InetAddress.getLocalHost()` is used. |
| `localPort` | Port the `DiscoverySpi` listens to | `47500` |
| `localPortRange` | Local port range. Local node tries to bind on the first available port starting from local port until local port plus local port range | `100` |
| `heartbeatFrequency` | Delay in milliseconds between heartbeat issuing of heartbeat messages. SPI sends messages in configurable time interval to other nodes to notify them about its state. | `2000` |
| `maxMissedHeartbeats` | Number of heartbeat requests that could be missed before local node initiates status check. | `1` |
| `reconnectCount` | Number of times node tries to (re)establish connection to another node. | `2` |
| `networkTimeout` | Sets maximum network timeout in milliseconds to use for network operations. | `5000` |

| Property Name | Description | Default Value |
|---|---|---|
| `socketTimeout` | Sets socket operations timeout. This timeout limits the connection time and write-to-socket time. | `2000` |
| `ackTimeout` | Sets timeout for receiving acknowledgment for sent message. If acknowledgment is not received within this timeout, sending is considered as failed and SPI tries to repeat message sending. | `2000` |
| `joinTimeout` | Sets join timeout. If non-shared IP finder is used and node fails to connect to any address from IP finder, node keeps trying to join within this timeout. If all addresses are still unresponsive, an exception is thrown and the node startup fails. The zero (0) value indicates that wait forever. | `0` |
| `threadPriority` | Thread priority for threads started by SPI. | `0` |
| `statisticsPrintFrequency` | Statistics print frequency in milliseconds. The zero (0) value indicates that no print is required. If the value is greater than zero (0) and the log is not quiet, then statistics are printed with INFO level once a period. This might be helpful for tracing the topology problems. | `0` |

### Example: Multicast based Discovery

`TcpDiscoveryMulticastIpFinder` uses multicast to discover other nodes with the same `multicastGroup` in the grid and is the default IP finder.

```
<bean class="org.apache.ignite.configuration.IgniteConfiguration">
  ...
  <property name="discoverySpi">
    <bean class="org.apache.ignite.spi.discovery.tcp.TcpDiscoverySpi">
      <property name="ipFinder">
        <bean 
class="org.apache.ignite.spi.discovery.tcp.ipfinder.multicast.TcpDiscoveryMulticastI
pFinder">
          <property name="multicastGroup" value="228.10.10.157"/>
        </bean>
      </property>
    </bean>
  </property>
</bean>
```

### Example: Static IP based Discovery

For cases when multicast is disabled, use `TcpDiscoveryVmIpFinder` with pre-configured list of IP addresses.

```
<bean class="org.apache.ignite.configuration.IgniteConfiguration">
  ...
  <property name="discoverySpi">
    <bean class="org.apache.ignite.spi.discovery.tcp.TcpDiscoverySpi">
      <property name="ipFinder">
        <bean
class="org.apache.ignite.spi.discovery.tcp.ipfinder.vm.TcpDiscoveryVmIpFinder">
          <property name="addresses">
            <list>
              <!--
     Explicitly specifying address of a local node to let it start and operate
normally even if there is no more nodes in the cluster.You can also optionally
specify an individual port or port range.
      -->
              <value>1.2.3.4</value>

              <!--
                  IP Address and optional port range of a remote node. You can also
optionally specify an individual port and don't set    the port range at all.
              -->
              <value>1.2.3.5:47500..47509</value>
            </list>
          </property>
        </bean>
      </property>
    </bean>
  </property>
</bean>
```

### Example: Multicast and Static IP based Discovery

You can use both, multicast and static IP based discovery together. In this case, in addition to addresses received through multicast, if any, `TcpDiscoveryMulticastIpFinder` can also work with pre-configured list of static IP addresses, similar to static IP based discovery.

```
<bean class="org.apache.ignite.configuration.IgniteConfiguration">
  ...
  <property name="discoverySpi">
    <bean class="org.apache.ignite.spi.discovery.tcp.TcpDiscoverySpi">
      <property name="ipFinder">
        <bean
class="org.apache.ignite.spi.discovery.tcp.ipfinder.multicast.TcpDiscoveryMulticastI
pFinder">
          <property name="multicastGroup" value="228.10.10.157"/>

          <!-- list of static IP addresses-->
          <property name="addresses">
            <list>
              <value>1.2.3.4</value>

              <!--
                  IP Address and optional port range.
                  You can also optionally specify an individual port.
              -->
              <value>1.2.3.5:47500..47509</value>
            </list>
          </property>
        </bean>
      </property>
    </bean>
  </property>
</bean>
```

## Monitor Apache Ignite

When TIBCO MDM starts, Apache Ignite is intialized. You can monitor it on application server console. It also shows the current snapshot of cache topology.

```
[14:21:46]    __________  ________________
[14:21:46]   /  _/ ___/ |/ /  _/_  __/ __/
[14:21:46]  _/ // (7 7    // /  / / / _/
[14:21:46] /___/\___/_/|_/___/ /_/ /___/
[14:21:46]
[14:21:46] ver. 1.9.0#20170302-sha1:a8169d0a
[14:21:46] 2017 Copyright(C) Apache Software Foundation
[14:21:46]
[14:21:46] Ignite documentation: http://ignite.apache.org
[14:21:46]
[14:21:46] Quiet mode.
[14:21:46]   ^-- To see **FULL** console log here add -DIGNITE_QUIET=false or "-v"
to ignite.{sh|bat}
[14:21:46]
[14:21:46] OS: Windows Server 2008 R2 6.1 amd64
[14:21:46] VM information: Java(TM) SE Runtime Environment 1.8.0_112-b15 Oracle
Corporation Java HotSpot(TM) 64-Bit Server VM 25.112-b15
[14:21:46] Configured plugins:
[14:21:46]   ^-- None
[14:21:46]
[14:21:46] Message queue limit is set to 0 which may lead to potential OOMEs when
running cache operations in FULL_ASYNC or PRIMARY_SYNC modes due to message queues
growth on sender and receiver sides.
[14:21:46] Security status [authentication=off, tls/ssl=off]
[14:22:09] Performance suggestions for grid 'MDMNODE' (fix if possible)
[14:22:09] To disable, set -DIGNITE_PERFORMANCE_SUGGESTIONS_DISABLED=true
[14:22:09]   ^-- Enable G1 Garbage Collector (add '-XX:+UseG1GC' to JVM options)
[14:22:09]   ^-- Set max direct memory size if getting 'OOME: Direct buffer memory'
(add '-XX:MaxDirectMemorySize=<size>[g|G|m|M|k|K]' to JVM options)
[14:22:09]   ^-- Disable processing of calls to System.gc() (add '-
XX:+DisableExplicitGC' to JVM options)
[14:22:09] Refer to this page for more performance suggestions: https://
apacheignite.readme.io/docs/jvm-and-system-tuning
[14:22:09]
[14:22:09] To start Console Management & Monitoring run ignitevisorcmd.{sh|bat}
[14:22:09]
[14:22:09] Ignite node started OK (id=5da32f64, grid=MDMNODE)
[14:22:09] Topology snapshot [ver=1, servers=1, clients=0, CPUs=2, heap=2.0GB]
```

You can monitor Apache Ignite by using the following ways:

- JMX: JMX cache MBean of TIBCO MDM can be used to monitor various cache metrics and operations. Additionally, Apache Ignite has its own JMX, the default port is `49112`. By default, Apache Ignite based statistics for each cache are disabled for performance reasons.
- Visor Commandline Interface: Apache Ignite provides a command line interface called Visor for monitoring.
- Logging: TIBCO MDM generates a separate file for the Apache Ignite log.
- REST API: Apache Ignite supports REST API and the same has been integrated with TIBCO MDM. It can be used to perform different operations, such as, read or write cache (from or to), execute tasks, get various metrics, and so on.

## Logging

A new log configuration category, the `Default Ignite Log` is added in Configurator. A separate rolling log file, `ignite.log` has been configured with the default logging level to `INFO`.

By default, the following information is displayed at the `INFO` level:

- Ignite initialization log: shows the version of Apache Ignite, all different cache initialization, and cluster and topology change with the nodes add and remove. To know the topology, refer to the following line, which is displayed in the log at the TIBCO MDM startup:

```
Topology snapshot [ver=1, servers=1, clients=0, CPUs=8, heap=2.0GB]
```

- Periodic node metric with metricsLogFrequency: monitor node load with CPU and memory utilization metrics
- Periodic topology metric with `TcpDiscoverySpi` and`statisticsPrintFrequency`: monitor any change in topology

## Using Visor Command Line Interface

Apache Ignite supports scriptable command line monitoring capability, which is called as Visor. By using Visor, you can get statistics about nodes, caches, and tasks in the grid and view the details about the topology showing various metrics and node configuration properties. Additionally, you can start and stop the remote nodes.

### Procedure

1. On the command line, type `$MQ_HOME/bin/ignite/bin/ignitevisorcmd.sh` or `%MQ_HOME%\bin\ignite\bin\ignitevisorcmd.bat`.
   The command line interface is started and `IGNITE_HOME` is automatically set to `$MQ_HOME/bin/ignite`.
2. To connect Visor to the grid, type *`visor> open -cpath=$MQ_HOME/config/IgniteMember.xml`*
   The `IgniteMember.xml` file is the cluster topology configuration file used with TIBCO MDM.
3. Type `help cmd` or `? cmd` to retrieve a complete list of commands.
   For the commonly used commands, refer to Visor Commands.

### Visor Commands

*Visor Commands*

The following are commonly used commands available in Visor:

| Command | Description |
|---|---|
| ack | Acknowledges arguments on all remote nodes |
| alert | Alerts for user-defined events |
| cache | Prints cache statistics, clears cache, and prints a list of all entries from cache |
| close | Disconnects Visor console from the grid |
| config | Prints node configuration |
| deploy | Copies file or folder to remote host |
| disco | Prints topology change log |
| events | Print events from a node |
| gc | Runs GC on remote nodes |

| Command | Description |
|---|---|
| help | Prints Visor console help |
| kill | Kills or restarts node |
| log | Starts or stops grid-wide events logging |
| mclear | Clears Visor console memory variables |
| mget | Gets Visor console memory variables |
| mlist | Prints Visor console memory variables |
| node | Prints node statistics |
| open | Connects Visor console to the grid |
| ping | Pings node |
| quit | Quits from Visor console |
| start | Starts or restarts nodes on remote hosts |
| status | Prints Visor console status |
| tasks | Prints tasks execution statistics |
| top | Prints current topology |
| vvm | Opens VisualVM for nodes in topology |

## Securing Connection among Nodes Using SSL

By using the SSL socket communication, you can secure connection among all nodes of Apache Ignite.

### Procedure

1. Navigate to `$MQ_HOME/config` and open the `IgniteMember.xml` file.
   a) Set `sslContextFactory`: by default, Apache Ignite provides a default SSL context factory, `org.apache.ignite.ssl.SslContextFactory`, which uses configured `keystore` to initialize SSL context.

```
<property name="sslContextFactory">
    <bean class="org.apache.ignite.ssl.SslContextFactory">
      <property name="keyStoreFilePath" value="keystore/keystore.jks"/>
      <property name="keyStorePassword" value="123456"/>
      <property name="trustStoreFilePath" value="keystore/truststore.ts"/>
      <property name="trustStorePassword" value="123456"/>
    </bean>
  </property>
```

   b) Disable Certificate Validation: in some cases, you must disable certificate validation of the client side. For example, when connecting to a server with self-signed certificate
   Set a disabled trust manager to `sslContextFactory`

```
<property name="sslContextFactory">
    <bean class="org.apache.ignite.ssl.SslContextFactory">
      <property name="keyStoreFilePath" value="keystore/keystore.jks"/>
      <property name="keyStorePassword" value="123456"/>
      <property name="trustManagers">
        <bean class="org.apache.ignite.ssl.SslContextFactory" factory-
method="getDisabledTrustManager"/>
     </property>
    </bean>
  </property>
```

c) Set Protocol: By using Apache Ignite, you can configure different types of encryption. The following algorithms are supported http://docs.oracle.com/javase/7/docs/technotes/guides/security/StandardNames.html#SSLContext and can be set by using the `setProtocol` method. The `TLS` encryption is the default.

```
<property name="sslContextFactory">
    <bean class="org.apache.ignite.ssl.SslContextFactory">
      <property name="setProtocol" value="SSL"/>
      ...
    </bean>
  </property>
  ...
```

2. Save the `IgniteMember.xml` file.

If security is configured, the logs contain `communication encrypted=on`.

```
INFO: Security status [authentication=off, communication encrypted=on]
```

The server console shows the following:

```
INFO: Security status [authentication=off, tls/ssl=on]
```

3. Generate `keyStore` using the following command:

```
keytool -genkey -alias ignite -keystore keystore.jks -keyalg RSA
```

4. Generate `trustStore` using the following two commands:

- `keytool -export -file ignite.cert -keystore keystore.jks -alias ignite`
- `keytool -import -v -trustcacerts -file ignite.cert -keystore truststore.ts -alias ignite`

For more information, see Apache Ignite documentation.

# Starting Apache Ignite As External Cache Server

You can start Apache Ignite as an external cache server in case of client-server configuration or centralized cache server topology.

### Prerequisites

Ensure that the `ECMClasses.jar` file located in `$MQ_HOME/lib/mq` is in the class path while starting external Apache Ignite cache server. By default,`ignite.bat` or `ignite.sh` handles this scenario by referring `$MQ_HOME` environment variable. However, if `$MQ_HOME` is not set, then copy `ECMClasses.jar` from the TIBCO MDM installation directory to the `$MQ_HOME/bin/ignite/libs` directory.

### Procedure

1. Clone the `IgniteMember.xml` configuration file. For example, `IgniteMember2.xml`
2. Navigate to `$MQ_HOME/bin/ignite/bin` and run either of the following command in the command prompt:

```
ignite.bat %MQ_HOME%\config\IgniteMember2.xml
```

Or

```
ignite.sh $MQ_HOME/config/IgniteMember2.xml
```

3. Start TIBCO MDM with Apache Ignite as a client or server by changing the `clientMode` value to `true` or `false` respectively.

# Configuration of Golden Record Cache and Cluster Topology with Apache Ignite

You can enable the golden record cache using the `FastCache enable` property in Configurator. For information on configuration properties of golden record cache, refer to the section, "Configuration Properties of Golden Record Cache" in *TIBCO MDM System Administration*.

To configure golden record cache and cluster topology with Apache Ignite, use the `FastCacheIgniteMember.xml` file, which is located at `$MQ_HOME/config`. The configuration is similar to the one specified in the `IgniteMember.xml` file. For information, refer to Configuring Cache and Cluster Topology for Apache Ignite.

Golden record cache with Apache Ignite has few limitations. For information, refer Limitations.

# Apache Ignite Durable Memory

Apache Ignire provides a page-based (durable) memory architecture that is split into pages of fixed size. The pages are stored in managed off-heap regions in RAM (outside of Java heap).

### Memory Configuration

By default, Apache Ignite nodes consume up to 2GB of the RAM available locally. You can change the size of the default memory by using the `totalSize` property in the `IgniteMember.xml` file. Specify the memory size in MB. For example,

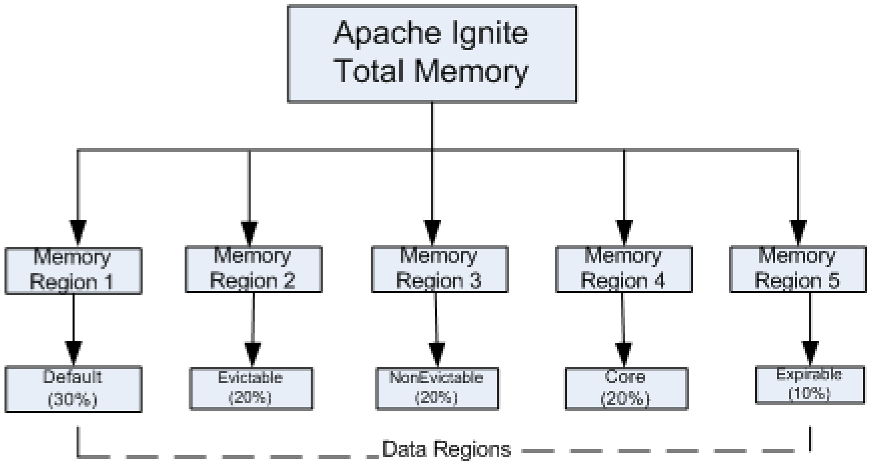
```
<property name="properties">
<util:properties>
<!-- Size in MB -->
<prop key="totalSize">2048</prop>
</util:properties>
</property>
```

### Data Region

A data region is a logical expandable area. Durable memory includes one or more data regions that can vary in size and eviction policies, be persisted on disk. By default, the Apache Ignite node creates the Default region that allocates 30% memory of total memory. The TIBCO MDM Ignite node creates the following custom types of region:

- Core: Allocates 20% memory of total memory
- Evictable: Allocates 20% memory of total memory
- NonEvictable: Allocates 20% memory of total memory
- Expirable: Allocates 10% memory of total memory

The following diagram represents the structure of Apache Ignite durable memory structure in TIBCO MDM:



## Limitations

The following features are not supported with Apache Ignite:

- Repository spaces
- Bundle Caching
- Golden record cache
  - Dynamic index creation, modification and deletion are not supported using REST services. You can create indexes on the server startup for a particular object type and the column name in the `FastCache_CacheConfig.xml` file.

    For cache and index configuration of golden record cache, refer to the section, "Cache Index using `FastCache_CacheConfig.xml`" in *TIBCO MDM System Administration*.
- CacheManger Utility: If the `Ignite` option is selected for the `Cache Type` (`com.tibco.cim.cache.type`) property in Configurator, the `CacheManager.bat` or `CacheManager.sh` utility does not work. Use the Visor command line utility to get statistics about nodes, caches, and tasks in the grid and view the details about the topology.

## Troubleshooting with Apache Ignite

If you encounter an error while working with Apache Ignite, you might resolve some of the common issues by completing the troubleshooting procedures.

| Issue | Solution |
|---|---|
| Multi-node cache cluster does not connect. | • Ensure that you have correct configuration for discovery for `discoverySpi`. That means, ensure that correct and all required IP addresses and port are defined.<br>• Default configuration has restricted node discovery to loop-back address. To enable the discovery of remote nodes, remove the property `<property name="localHost" value="127.0.0.1"/>` from the `IgniteMember.xml` file. |

| Issue | Solution |
|---|---|
| Command line interface does not connect or does not show all nodes in topology. | Supply the configuration file while running CLI tool. Ensure to use the same `IgniteMember.xml` configuration file, which is used to start TIBCO MDM. |

# Apache Spark

Apache Spark is a high-performance engine for large-scale data processing. Use Apache Spark in TIBCO MDM to process huge data loads. Apache Spark improves the big data import performance of TIBCO MDM as it scales well across the nodes and recovers well from failures.

- For information about Apache Spark, see Apache Spark documentation.
- For information about using the big data import feature, see the "Importing of Records Using Big Data Import" section in *TIBCO MDM User's Guide*.

## Platform Limitations For Apache Spark

The following table lists the supported and unsupported platform combination.

### Supported Platform Combination

Apache Spark works with the following platform combinations:

| Operating System | Application Server | Database |
|---|---|---|
| Linux | JBoss WildFly | Oracle |
| Linux | JBoss EAP | Oracle |
| Linux | JBoss WildFly | Microsoft SQL Server |
| Linux | JBoss WildFly | PostgreSQL |

For the supported platforms, Apache Spark works with the Apache Ignite cache.

### Unsupported Platform Combination

Apache Spark does not work with the following platform combinations:

| Operating System | Application Server | Database |
|---|---|---|
| Linux | WebSphere | Oracle |
| Linux | WebLogic | Oracle |

Windows operating system with any application server and database is not supported.

## Setting up Apache Spark

For the Big Data Import feature of TIBCO MDM, you must download and configure Apache Spark. Apache Spark cluster includes a single master and any number of worker nodes. For high and efficient performance, configure four or five worker nodes.

### Prerequisites

- For the recommended platform, see Platform Limitations for Apache Spark.
- Share the `$MQ_HOME`, `$MQ_COMMON_DIR`, and `MQ_CONFIG_FILE` directories across all worker nodes (from TIBCO MDM host machine to Apache Spark master and worker machines).

For information about sharing the directory in the cluster environment, see Clustering Set Up.

### Procedure

1. **Add Entries in the Hosts File (master and worker)**
   1. On the Apache Spark master machine, navigate to `/etc`.
   2. In the `hosts` file, specify names of the servers individually for all three nodes.

      Consider a scenario where you have three servers: one for the master node, and two others for the worker node. In other words, these are three Linux machines where Apache Spark is already installed.

      ```
      spark1: master node
      spark2: worker node
      spark3: worker node
      ```

      Repeat the step a and b for all worker node machines.

      Ensure that all the servers can ping each other by name and an IP address.
2. **Generate Key**
   1. Generate the key from http://www.thegeekstuff.com/2008/11/3-steps-to-perform-ssh-login-without-password-using-ssh-keygen-ssh-copy-id/.

      You need the key to access worker nodes without password. If you have generated the key, you can seamlessly copy the key to the worker node machines without password.
3. **Setup SSH Passwordless Login**
   1. On the command line, enter the following command to copy the key to the worker node machines:

      ```
      ssh-keygen

      ssh-copy-id -i ./.ssh/id_rsa.pub spark1
      ssh-copy-id -i ./.ssh/id_rsa.pub spark2
      ssh-copy-id -i ./.ssh/id_rsa.pub spark3
      ```
4. **Check Login to Worker Nodes using SSH Keys**
   1. Enter the following command from the master node and verify if the worker nodes are accessible.

      ```
      ssh 10.XXX.XXX.XXX (an IP address of the worker node)
      ```

   You can now log in to the worker node machine without being prompted for a password.
5. **Install Spark**
   1. To download Apache spark, navigate to the Apache Spark website.
   2. In the **Archived Releases** section, click Spark Release archives.
   3. From the version list, click the spark-2.2.0 version.
   4. Download **spark-2.2.0-bin-hadoop2.7.tgz** and extract its contents to the `/home/`*`username`* folder.
6. **Set up Apache Spark Cluster**
   1. If the `slaves` file does not exist, copy `slaves.template` to the `slaves` directory.
   2. Navigate to `$SPARK_HOME/conf` and list the following slave or worker nodes in the `slaves` file:

      - `spark1`
      - `spark2`
      - `spark3`
7. **Configure Spark Master**
   1. If the `spark-env.sh` file does not exist, copy `spark-env.sh.template` to `spark-env.sh`.
   2. Navigate to `$SPARK_HOME/conf` and open the `spark-env.sh` file.

c) Add the IP address and port number of the `SPARK_MASTER` node and instance number of the `SPARK_WORKER` node:

```
SPARK_MASTER_IP=10.XXX.XXX.XXX (an IP address of the master node)
SPARK_MASTER_PORT=7077
SPARK_WORKER_INSTANCES=1
```

d) Set `MQ_LOG` in the `spark-env.sh` file.

> Before setting `MQ_LOG` parameter, ensure that `MQ_CONFIG_FILE` is shared to Apache Spark master node machine and has full permissions. If Apache Spark cluster and TIBCO MDM are installed on different machines, share `$MQ_HOME`. For this, you must share `$MQ_HOME` from TIBCO MDM host machine to Apache Spark master and worker machines.

e) Set `JAVA_HOME` in the `spark-env.sh` file.

```
export JAVA_HOME=/home/apps/JAVA8/jdk1.8.0_112
export PATH=$PATH:$JAVA_HOME/bin
```

f) Copy `$SPARK_HOME` folder (that is, `spark-2.2.0-bin-hadoop2.7`) from `/home/`*`username`*`/` to all the other worker node machines.

8. **Start Apache Spark Cluster**

a) On the master node, run **`start-all.sh`** from`/home/`*`username`*`/spark-2.2.0/sbin`.
The master and worker nodes start.

b) Optional: Navigate to `$SPARK_HOME/sbin` folder and run the following command to stop Apache Spark: **`stop-all.sh`**

9. **Check Whether Services Have Been Started**

a) Run the **`JPS`** command from any path to list the processes (master and slave).
Running **`JPS`** from master machine shows master node processes and slave machines shows worker nodes processes.

```
username@TIBCO MDM master machine name cd $SPARK_HOME/sbin
[username@TIBCO MDM master machine name sbin]# jps
5495 Master
5607 Jps
```

```
username@TIBCO MDM worker machine name cd $SPARK_HOME/sbin
[username@TIBCO MDM worker machine name sbin]# jps
1234 Worker
5678 Jps
```

Apache Spark is set up.

### What to do next

To verify if Apache Spark setup works correctly, run the samples provided in the `$SPARK_HOME/examples/src/main/scala/org/apache/spark/examples` directory.

## Accessing Apache Spark User Interface

You can access the Apache Spark master, worker, application UI to know about configured worker nodes, cluster resources, and spark jobs.

### Prerequisites

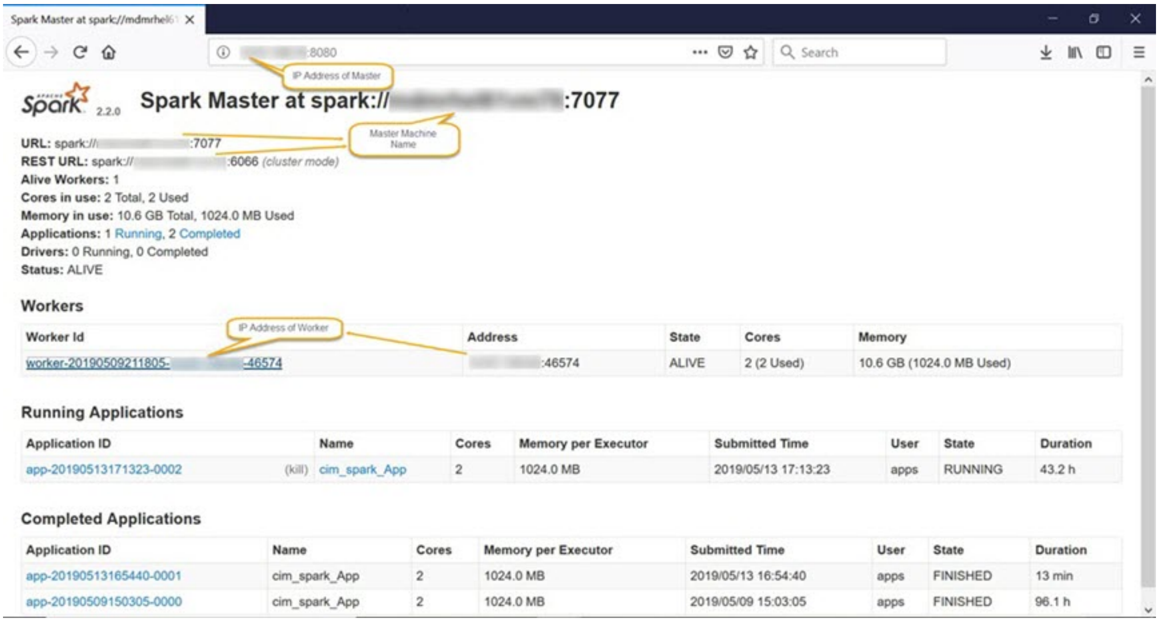
Set up Apache Spark.

### Procedure

1. **Apache Spark Master Web UI**

Browse the Apache Spark UI to know about worker nodes, running application, and cluster resources.

a) Run the following URL to access the Apache Spark master UI: `http://`*`IP Address of Master`*`:8080/`



Spark 2.2.0 **Spark Master at spark://**[redacted]**:7077**

**URL:** spark://[redacted]:7077
**REST URL:** spark://[redacted]:6066 *(cluster mode)*
**Alive Workers:** 1
**Cores in use:** 2 Total, 2 Used
**Memory in use:** 10.6 GB Total, 1024.0 MB Used
**Applications:** 1 Running, 2 Completed
**Drivers:** 0 Running, 0 Completed
**Status:** ALIVE

**Workers**

| Worker Id | Address | State | Cores | Memory |
|---|---|---|---|---|
| worker-20190509211805-[redacted]-46574 | [redacted]:46574 | ALIVE | 2 (2 Used) | 10.6 GB (1024.0 MB Used) |

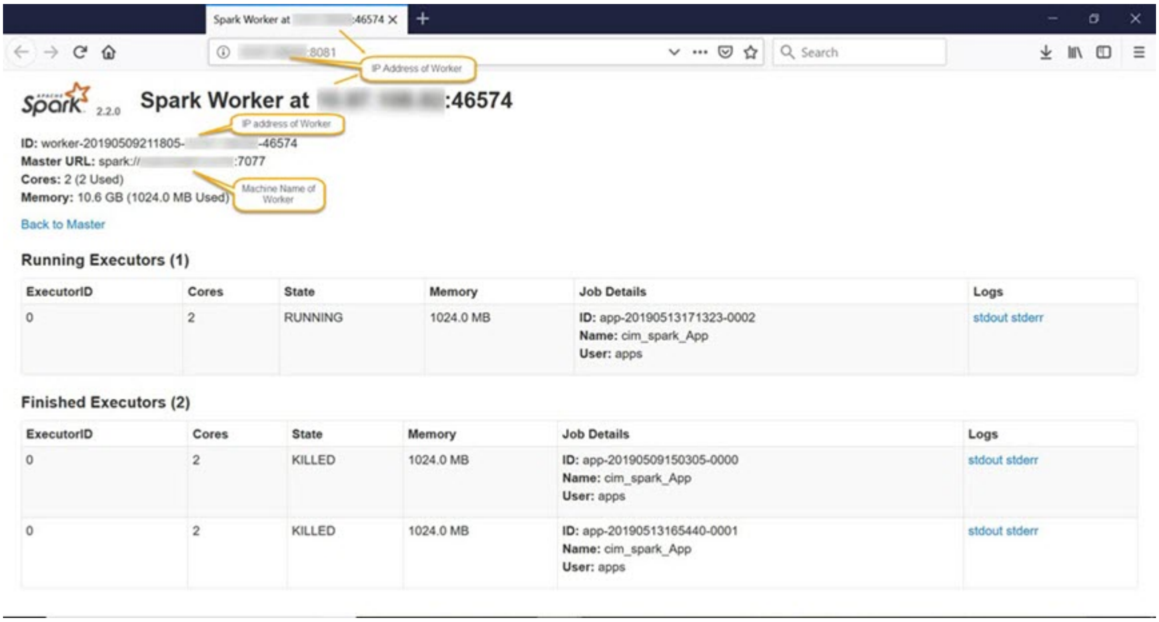
**Running Applications**

| Application ID | Name | Cores | Memory per Executor | Submitted Time | User | State | Duration |
|---|---|---|---|---|---|---|---|
| app-20190513171323-0002 (kill) | cim_spark_App | 2 | 1024.0 MB | 2019/05/13 17:13:23 | apps | RUNNING | 43.2 h |

**Completed Applications**

| Application ID | Name | Cores | Memory per Executor | Submitted Time | User | State | Duration |
|---|---|---|---|---|---|---|---|
| app-20190513165440-0001 | cim_spark_App | 2 | 1024.0 MB | 2019/05/13 16:54:40 | apps | FINISHED | 13 min |
| app-20190509150305-0000 | cim_spark_App | 2 | 1024.0 MB | 2019/05/09 15:03:05 | apps | FINISHED | 96.1 h |

2. **Apache Spark Worker Web UI**
   a) Run the following URL to access the Apache Spark worker UI: `http://`*`IP Address of Master`*`:8081/`



Spark 2.2.0 **Spark Worker at** [redacted]**:46574**

**ID:** worker-20190509211805-[redacted]-46574
**Master URL:** spark://[redacted]:7077
**Cores:** 2 (2 Used)
**Memory:** 10.6 GB (1024.0 MB Used)
Back to Master

**Running Executors (1)**

| ExecutorID | Cores | State | Memory | Job Details | Logs |
|---|---|---|---|---|---|
| 0 | 2 | RUNNING | 1024.0 MB | **ID:** app-20190513171323-0002<br>**Name:** cim_spark_App<br>**User:** apps | stdout stderr |

**Finished Executors (2)**

| ExecutorID | Cores | State | Memory | Job Details | Logs |
|---|---|---|---|---|---|
| 0 | 2 | KILLED | 1024.0 MB | **ID:** app-20190509150305-0000<br>**Name:** cim_spark_App<br>**User:** apps | stdout stderr |
| 0 | 2 | KILLED | 1024.0 MB | **ID:** app-20190513165440-0001<br>**Name:** cim_spark_App<br>**User:** apps | stdout stderr |

## Setting up Hadoop Distributed File System

Apache Spark is compatible with Hadoop data. You can run it in Hadoop clusters through YARN or Apache Spark's standalone mode, and it can process data in Hadoop Distributed File System (HDFS). HDFS is highly fault tolerant and efficient in parallel data processing. HDFS takes in data, breaks the information into separate blocks, and distributes them to different nodes in a cluster.

For more information on HDFS, see HDFS documentation.

### Prerequisites

Create `hdfs/namenode` and `hdfs/datanode` directories at the `home/`*username*.

For example, `home/`*`username`*`/hadoop/hdfs/namenode` and `home/`*`username`*`/hadoop/hdfs/datanode`.

### Procedure

1. Download Apache Hadoop 2.7.3 version from https://hadoop.apache.org/releases.html and extract its contents to the `/home/`*`username`* folder.
2. Navigate to `$HADOOP_HOME/etc/hadoop` and modify the `hdfs-site.xml` file to set the path for `namenode` and `datanode`.

```
<configuration>
 <property>
   <name>dfs.namenode.name.dir</name>
     <value>/home/username/hadoop/hdfs/namenode</value>
       </property>
 <property>
    <name>dfs.datanode.data.dir</name>
      <value>/home/username/hadoop/hdfs/datanode</value>
        </property>
</configuration>
```

3. Modify the `core-site.xml` file to specify the HDFS host name and port.

```
<configuration>
<property>
    <name>fs.default.name</name>
        <value>hdfs://hostname</value>
          </property>
</configuration>
```

4. Modify the `Hadoop-env.sh` file to set `JAVA_HOME`.
5. Navigate to `$HADOOP_HOME/sbin` directory and run `start-dfs.sh`.
   The HDFS setup is completed and the HDFS processes are started.

   Optional: If you want to stop Apache Hadoop, run `stop-dfs.sh`.

### What to do next

- To verify if HDFS is running correctly , run the **`JPS`** command from any path. The following processes with ID are displayed:

```
[username@TIBCO MDM master machine name sbin]# jps
5607 Jps
4634 DataNode
4842 SecondaryNameNode
5132 NodeManager
4527 NameNode
5023 ResourceManager
```

- Configure TIBCO MDM with Apache Spark. For information, see the "Configuration Properties for Apache Spark" and "Required JAR Files for Apache Spark" sections in *TIBCO MDM User's Guide*.

# Clustering Set Up

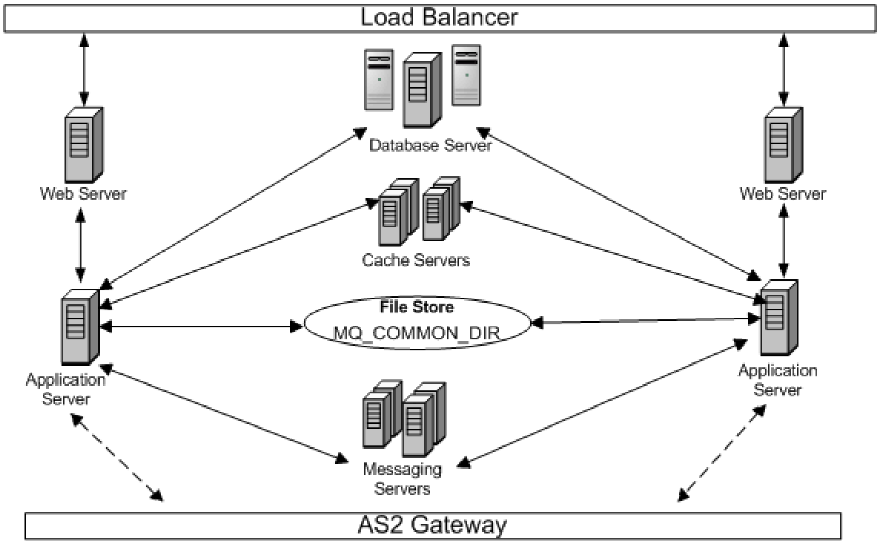
TIBCO MDM is designed for deployment in a cluster for the purpose of load balancing and redundancy. Multiple components of the application can be run as multiple instances in the cluster. You can deploy TIBCO MDM in a cluster environment.

## Clustering Architecture and Components

Certain components of TIBCO MDM can be run as multiple instances to share the load, and can provide redundancy, whereas other components must be ran as one instance.

The following diagram depicts the clustering architecture of TIBCO MDM.

*Clustering Architecture*



*Clustering Components*

| Clustering Component | Description |
|---|---|
| Load Balancer | To deploy clustered web servers, use a load balancer. A load balancer equally distributes HTTP requests from the browser and web service requests to the cluster members. This guide does not describe how to set up the load balancer. Consult the appropriate vendor documentation.<br>Clustering of web servers is optional, and if not used, a load balancer is not required. |
| Web Server | Web Servers receives HTTP requests and forward them to the application server. As mentioned earlier, you need a load balancer to cluster web servers.<br>A single web server can be set up to load balance the HTTP requests to multiple application servers without any load balancer. |

| Clustering Component | Description |
|---|---|
| Application Server | You can install one or more instances of the TIBCO MDM application on one computer running an application server, or install an instance of the TIBCO MDM application each running in an application server running on a different machine.<br>For clustering, multiple application servers *must* be deployed for load balancing and to provide redundancy. All application servers in the cluster *must* have the same JVM version and have compatible JVMs ensure that:<br>• Each application server has an independent JNDI registry.<br>• Each server has a unique port assigned for JNDI registry in the Configurator.<br>• Each application server has its own logging setup, with separate logging configuration, and a setup where the log files are located in a directory on a local file system.<br>• The configuration is centrally stored, where each application server instance pulls its configuration information from the central cluster configuration instance. The cluster configuration instance is referred to in the `MQ_CONFIG_FILE` environment variable and typically points to a file named `ConfigValues.xml`. Each application instance can pull the relevant configuration information out of the centrally configured configuration by identifying itself through its unique node ID.<br>• The Node ID (or `NODE_ID` environment variable) is set uniquely for each application server instance and matches the member name in the Configurator.<br>For cluster configuration with JBoss, refer to the following link: https://docs.jboss.org/author/display/AS71/AS7+Cluster+Howto |
| Database Server | A Database server persists and queries the TIBCO MDM data. All the application servers in the cluster *must* be connected to *one active* database instance.<br>**Note:** If you need to cluster the Oracle database using RAC, contact Customer Support.<br>If the application server (for example, WebSphere) supports transparent failover between active and standby database servers, the TIBCO MDM is able to connect to the standby database server. Any industry standard database clustering technology can be used to cluster databases. In the case of a database failover and restart, application servers are able to reconnect to the database without requiring a restart. |

| Clustering Component | Description |
|---|---|
| Messaging Server | A Messaging server is used for internal application server synchronization purposes as well as external communication with backend systems.<br>All application servers should be connected to at least one active messaging server (for example, TIBCO EMS). The messaging servers themselves may be clustered. To configure clustering, refer to the relevant document for messaging servers.<br>Multiple standby messaging servers *may* be configured using the messaging configuration inside the `ConfigValues.xml` configuration instance. When the primary messaging server fails, all open connections to the server are transparently routed to the standby server. During the reconnection phase to the standby server, the TIBCO MDM server can encounter errors. However, typically the rollover operation to the standby server executes quickly.<br>If the messaging server goes down, the application servers can be configured to attempt reconnection to the messaging server for a certain configurable interval. After that time frame, the application server has to be restarted<br>**Note**: It is possible to configure the application in such a way that different instances can use a segregated, dedicated JMS server. This configuration may be used to create prioritized processing zones. Consult Customer Support for additional information.<br>If WebSphere MQ is used as the messaging server, the number of JMS sessions that can be created needs to be increased. This can be done by adding the following CHANNELS section to the qm.ini file that exists for Queue Manager used by the cluster (for example, on Linux or UNIX machines, qm.ini might exist in the /var/mqm/qmgrs/<QMgrName> directory).<br>CHANNELS:<br>MaxChannels=400<br>(or later, depending on the number of channels) |

| Clustering Component | Description |
|---|---|
| File Stores | File Stores are described in detail as follows:<br>• The MQ_COMMON_DIR directory is shared by all application servers. Ensure that all servers are set up to point to the same location: MQ_COMMON_DIR. The location can be mapped to a different logical directory name for each server.<br>For example, one application server can mount MQ_COMMON_DIR to /home/mdm/common, and another one can mount MQ_COMMON_DIR to /export/vsamin/commondir. In addition, a Communicator running on its own machine can mount MQ_COMMON_DIR to /mdm6/commondir, provided all of them point to the same physical file store.<br>• The MQ_HOME directory can be set up in any one of the following ways:<br>– Each application server has its own MQ_HOME and it is not shared with other application servers.<br>– MQ_HOME is shared for all application servers. This will typically involve a single install image for TIBCO MDM, which is shared throughout the cluster machines through a remote file system.<br>• The MQ_CONFIG_FILE file represents the central configuration store for the entire cluster, containing the configuration for every instance. In order to set up the logging configuration for each cluster member, define the cluster in the Configurator, and define for each member the relevant logging configuration in **Member** > **Logging**. Also, define the MQ_LOG environment variable in the application server startup script so that it points to a directory in a local file system<br>**Note**: You can configure the message recovery system to write failed messages to a local file system or a network file system. For more details, refer to *TIBCO MDM System Administration.* |

# Concurrent Process Synchronization in a Clustered Environment

TIBCO MDM has two timer tasks (MqRevivify and FileWatcher) which repeat execution after a specified interval. There can be contention issues when multiple task threads run concurrently in the cluster.

To ensure that such issues do not occur, TIBCO MDM implements a locking mechanism by which when one timer task is running, other tasks either skip the processing or do not work on the same file (as in the case of FileWatcher).

### Prerequisites

Each application instance must have a unique Node ID. A Node ID is a logical name for the server instance.

It is recommended to use the *hostname_serverinstancenumber* naming convention for the NODE_ID . For example, on the lightning machine the server instance is 1. In this case, the NODE_ID is lightning_1.

- For server instance 1, NODE_ID=lightning_1
- For server instance 2, NODE_ID=lightning_2

Specify NODE_ID in the standalone.xml file under <system-properties> element. The file is located at $JBOSS_HOME\standalone\configuration directory. For example,

```
<system-properties>
<property name="NODE_ID" value="Member1"/>
</system-properties>
```

It is important that each server instance has a different Node ID value and that the Node ID value matches the name in the server instance in the Configurator.

## Locking Mechanism

The locking mechanism implementation process is explained in this topic.

### Procedure

1. When a timer task starts processing, it tries to acquire a lock.
2. If it finds a lock file, it skips processing.
3. If a lock file is not found, a lock file is created and the Node ID information of the server is added to the lock file. The lock files are created in the following directories:

    - MqRevivify: $MQ_HOME/Work/MqRevivify.lock
    - FileWatcher: Location can be specified in the FileWatcher.xml file.

        If multiple files are being processed (using Filewatcher), ensure the lock file is enabled in the filewatcher.xml file (enabled by default).

    The lock file contains the local host information so that the host JVM that created it can be identified. It contains the application Node ID and host identifier. This needs to be done so that only the owner can purge the file later.

4. After the task is completed, the lock file is deleted.

## Example FileWatcher

FileWatcher scans a file and sends it for processing. After the file is processed, the file is moved to the done folder. If more than one FileWatcher threads attempt to process the file, concurrency issues may occur.

To prevent concurrency issues:

- Before FileWatcher picks up a file for processing, a lock is placed on the parent directory. This way, only one FileWatcher can pick up files from one directory. The FileWatcher locking synchronizes access to a directory for multiple FileWatcher threads.
- After the file is processed, the file lock is released.
- When the FileWatcher thread starts, it checks for any abandoned lock files for the available datasets. For example, lock files may have been left abandoned, as in the following cases:
    - The executing thread creates a lock file for a file.
    - The application terminates before the thread has completed processing.

## Deletion of Abandoned Lock Files

When a thread dies before a lock is released, it could result in abandoned lock files. Purging such lock files is necessary. In a clustered setup, only the cluster member server that created the lock can purge it later.

The Node ID information added to the lock file when it was created is used when purging the abandoned locks during server startup time.

The server checks for an existing lock file. If the lock file exists, the server opens it, reads the Node ID information, and determines whether it is the same instance on the same host who created the lock file. If they are same, the server assumes that the lock was abandoned and deletes the lock file.

The abandoned lock can only be removed when the correct server comes back up again. During the cluster member downtime other cluster members cannot perform the scheduled task.

The lock files created can be deleted manually from the following directories:

- Revivifier: `$MQ_HOME/Work/MqRevivify.lock`
- FileWatcher: Location specified in the `FileWatcher.xml` file.

## Fault Tolerant Messaging Using EMS

You can set up the fault tolerant of the TIBCO MDM messaging system for a single server or cluster using the TIBCO EMS messaging software. The EMS fault tolerant setup consists of a primary EMS server and a standby or backup server.

The two servers share the data store (typically on the file system) containing client information and the messages information. Initially, the primary server is active and the backup server monitors the primary server. When the primary server (or the host machine of that server) fails, the backup server will detect this and will become active. The messaging client (TIBCO MDM) also detects that failure and will transparently reconnect to the now active backup server.

Both the TIBCO MDM cluster and the EMS messaging software have to be configured for this deployment scenario.

### EMS Server Setup

Refer to the "Configuring Fault-Tolerant Servers" section from the *TIBCO EMS User's Guide* for details.

The configuration of the two EMS servers as a fault tolerant cluster involves configuring both message server configuration files (for example: *EMS-Configuration*`/tibco/cfgmgmt/ems/data/tibemsd.conf`). Both server names (property called server) have to be equal since they represent the same server. The configuration entry for fault tolerance involve the properties starting with `ft_*`. The most important one is ft_active, which will point to network address of the other message server.

The other values (ft_heartbeat, ft_activation, ft_reconnect_timeout) can be left at default values. During setup, the primary server should be started first and then the backup server. The backup server should print a message similar to `Server is in standby mode for tcp://myhost:7222`.

### TIBCO MDM Setup

The EMS Cluster has to be registered with TIBCO MDM. This is achieved by having multiple entries separated by a comma in the Cluster Server List property for both the Bus (Topic) and Queue setup.

In addition to the primary EMS server (`LocalhostServer`), you need to add the second server (`Server2`) to the Cluster Server List configuration value, which serve as a backup server. You must define the Server2 Server Connection String and Server2 Server encoding properties using **Add New Property.** Both have to be string values.

Also, check the Failed Connection Refresh Flag and Failed Connection Replace Optimization flags properties set as `true`.

Each TIBCO MDM server will retry several times to reconnect to the backup server in case of a failure. Choose `6` connection retry attempts (Failed Connection Retry Count) and `10000 ms` (or 10 seconds) time delay between attempts. These values work well with the default EMS cluster setup. The delay should not be less than `10s`. Choose the same value for ft_activation in the EMS cluster setup. Also the total time the TIBCO MDM Server attempts to reconnect to the backup server (6 * 10seconds = 60 seconds) will not be useful if it exceeds the ft_reconnect_timeout (by default 60 seconds).

Copy the *EMS-Configuration*`\tibco\cfgmgmt\ems\data` folder and rename the file, for example, rename it to data_secondary, and use it for the secondary server.

The same procedure has to be repeated for the TIBCO EMS Queue setup at **InitialConfig** > **Queue Setup** > **Cluster** > **TIBCO EMS.**

### Queue and Topic Setup

For Queue and Topic Setup, see "Configuring Queues and Topics" section of *TIBCO MDM System Administration*.

## Testing Clustered Installation

For testing clustered installation, first test the application installation, test the application for load balancing, and test the application and cache server for failover.

### Procedure

1. Testing the Application Installation
   a) Log in to TIBCO MDM.
   b) Create an enterprise.
   c) Create a repository in the specified enterprise.
   d) Upload data into the repository.
2. Testing the application for load balancing
   a) Verify that clones are working individually.
   b) Start the web browser and verify the following URLs:

   -
     ```
     http://NodeB:<port#>/eml/Home
     http://NodeC:<port#>/eml/Home
     http://NodeB/eml/Home
     ```

     (The IBM HTTP Server was installed on this node.)
   - You can now check the load balancing for the TIBCO MDM cluster.

   a) Access TIBCO MDM using the http://NodeB/eml/Home URL from different browsers and with different user accounts.
   b) Observe the details regarding which server receives the request and how both servers are loaded by viewing the http_plugin.log file located on Node B.
3. Testing the Application for Failover Without Session
   a) Access TIBCO MDM from two different browsers by entering the http://NodeB/eml/Home URL. Both the servers, for example, TIBCO MDM1 and TIBCO MDM2 serve one request each.
   b) On the first browser (assuming this request is being served by TIBCO MDM1), click **Add record for master catalog**. Enter the product ID, short description, and other information. Stop the TIBCO MDM1 server from Deployment Manager and then click **Save**. TIBCO MDM2 server's Login screen is displayed.
   c) Enter your **user ID** and **password** to continue working with TIBCO MDM.

   ```
   <UriGroup Name="default_host_veloselcluster_URIs">
           <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
   Name="/eml/*" />
           <Uri AffinityCookie="JSESSIONID" AffinityURLIdentifier="jsessionid"
   Name="/emlservlet/*" />
   </UriGroup>
   ```
4. Testing Failover of Cache Server
   a) Set up the Cache server on an external box. For more information on the Cache server, see Configuration of TIBCO MDM with TIBCO ActiveSpaces.
   b) Ensure the Cache server parameters are set in the application server startup scripts.

c) Add the activity to the list of sleep activities and set the sleep time. You can do this using the Failover Setup category in Configurator.

d) Add a record through the UI to initiate a workflow.

e) While the activity is sleeping, shut down the cache server that is running on the external box.

f) The Elink log shows a Cache server related error.

g) The Elink log also shows that the activity is retrying according to the parameters configured in the Configurator.

h) Bring up the Cache server while the activity is retrying. The workflow should be executed successfully.

# Support for Language Pack

TIBCO MDM supports the following languages for translation:

- French
- German
- Japanese
- Latin Spanish
- Simplified Chinese

The language packs are available at the TIBCO eDelivery site(https://edelivery.tibco.com) under the product, TIBCO MDM. A language pack is a JAR file, which contains translations of the TIBCO MDM GUI in a specific language. The JAR file contains all Java based resource bundles. The language pack file naming convention is `TIB_cim-lp_`*`releaseversion`*`_languagepack-`*`locale`*`.jar`. For example, the Simplified Chinese language pack for the TIBCO MDM release consists of a JAR file named `TIB_cim-lp_`*`releaseversion`*`_languagepack-zh-CN.jar`.

Download the JAR file of the language pack that you want to install. You can install the language pack using Configurator. For installation, refer to Installing Language Pack.

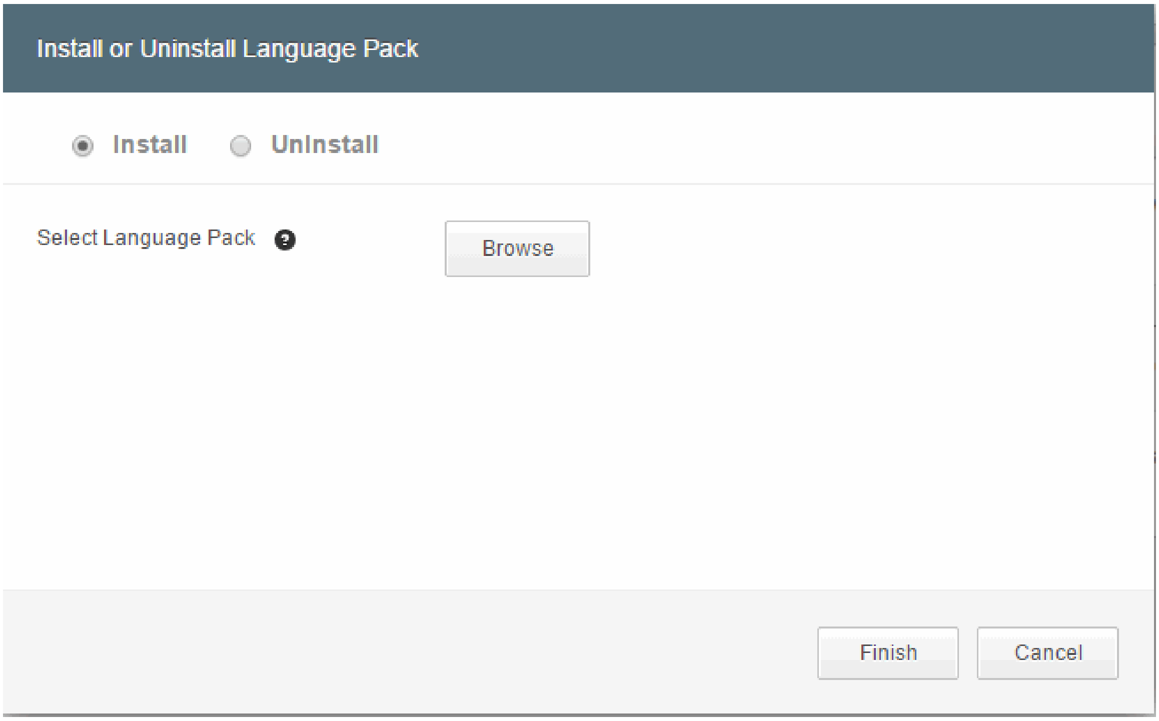
## Installing Language Pack

### Prerequisites

- Download the JAR file of the language pack that you want to install.
- Ensure that the following environment variables are properly set:
  - *$MQ_HOME*: The TIBCO MDM install location.
  - *$JAVA_HOME*: JDK 7 or later.
  - *$MQ_COMMON_DIR*: The TIBCO MDM Common directory.

### Procedure

1. Log in to Configurator.
2. Navigate to **Tools** > **Language Pack Installer**.
   The Install or Uninstall Language Pack window is displayed. By default, the **Install** option is selected.

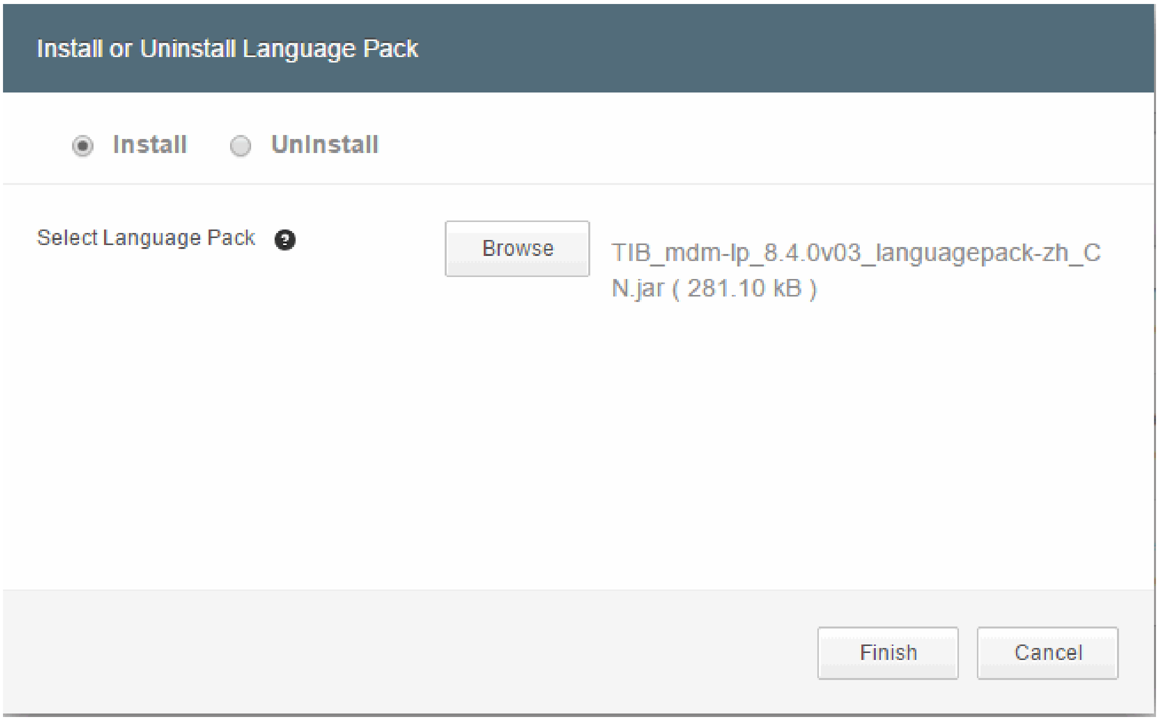


Install or Uninstall Language Pack

● Install ○ Uninstall

Select Language Pack

3. Click **Browse** and select the JAR file location of the language pack.
   The selected language pack name is displayed on the Install or Uninstall Language Pack window.



Install or Uninstall Language Pack

● Install ○ Uninstall

Select Language Pack TIB_mdm-lp_8.4.0v03_languagepack-zh_CN.jar ( 281.10 kB )

4. Click **Finish**.
   The `Language pack installed successfully` message is displayed.

   - The content of the JAR file is copied to the *$MQ_COMMON_DIR*`/Work/resources/tiblp.jar` file.

- The language code entry is made in the *$MQ_COMMON_DIR*/Work/languagePack/prop/lang.properties file.

> If you have installed JBoss WildFly Application Server, create the JBoss module as mentioned in Module Creation, and then perform the following tasks:
>
> - Copy the tiblp.jar file from *$MQ_COMMON_DIR*/Work/resources into the *$JBOSS_HOME*/modules/system/layers/base/com/tibco/mdm/maindirectory.
> - Open the module.xml file and add the following entry in the <resources> tag:
>
> ```
> <resource-root path="tiblp.jar"/>
> ```

### What to do next

Restart the application server manually, where TIBCO MDM is deployed.

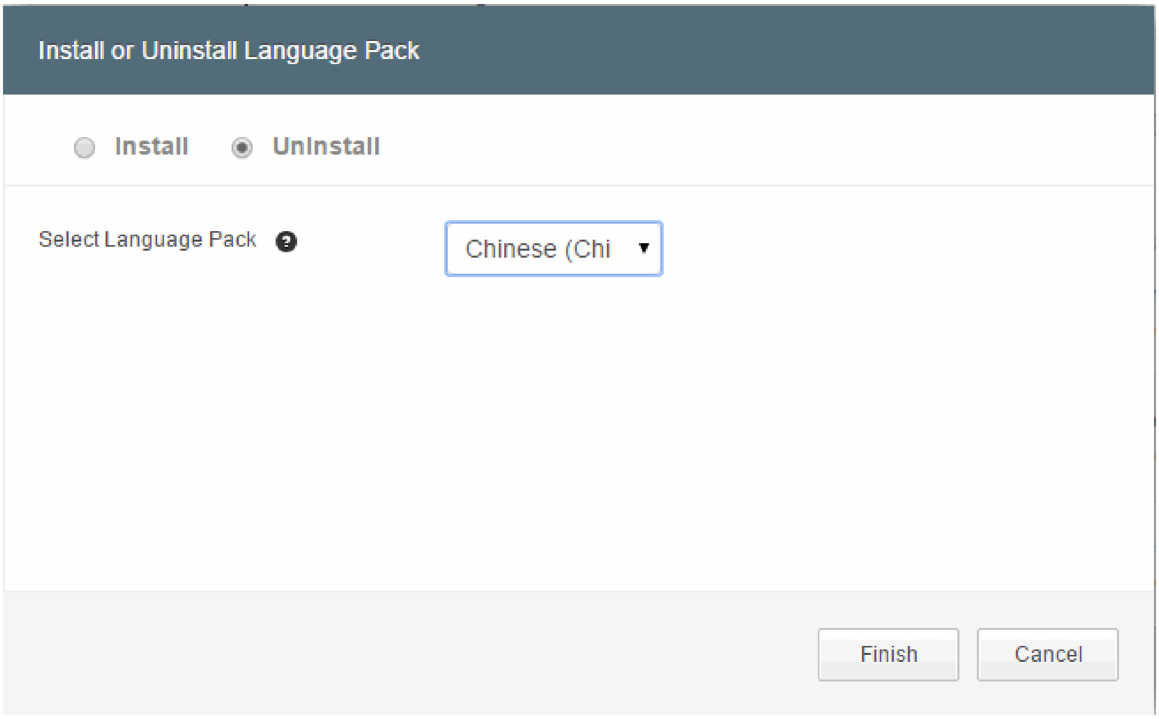
## Uninstalling Language Pack

### Prerequisites

Stop the application server.

### Procedure

1. Log in to Configurator.
2. Navigate to **Tools** > **Language Pack Installer**.
   The Install or Uninstall Language Pack window is displayed.
3. Select the **Uninstall** option.
   The installed languages are displayed in the **Select Language Pack** drop-down list.



4. Select the language that you want to uninstall.
5. Click **Finish**.
   The `Language pack uninstalled successfully` message is displayed.

- The property files of the selected language pack are deleted from the *$MQ_COMMON_DIR*/Work/resources/tiblp.jar file.
- The language code entry is removed from the *$MQ_COMMON_DIR*/Work/languagePack/prop/lang.properties file, that is, tiblp.jar is regenerated without deleting the resource bundles.

> For JBoss WildFly Application Server, copy the newly generated tiblp.jar to *$JBOSS_HOME*/modules/system/layers/base/com/tibco/mdm/main folder.

### What to do next

Restart the application server manually, where TIBCO MDM is deployed.

# GDSN Overview

Global Data Synchronization Network (GDSN) functionality is provided through a GDSN plugin component.

Most of the functionality between MDM and GDSN is common. The GDSN plug-in provides incremental functionality specific to GDSN. This plugin includes GDSN specific workflows, maps, rulebases and so on. GDSN specific screens and functionality will not be available with MDM edition.

You can use the following features of the GDSN Plugin:

- CIM to CIM Synchronization
- Creation of Enterprises of type Integration Hub, Retailer
- Default catalog formats and user accounts for a trading partner
- GDSN related Data (such as output maps and catalog formats specific to 1Sync and AS2)
- Incremental Resource Bundles

When the GDSN plugin is installed, resource bundles are not be switched automatically. This is to enable the MDM installation to continue using the terminology even if the GDSN plug-in is installed. You can switch the language bundle through the **Software Edition** property.

## Installing the GDSN Component

As per your requirement, you can install the GDSN plugin over the main MDM version of TIBCO MDM.

### Procedure

1. Extract the archive the product was delivered in.
2. Run the TIBCO UniversalInstaller application. This starts up the Universal Installer through which you can install TIBCO MDM GDSN Plugin.

   After a short delay while the installer initializes, the Welcome dialog is displayed. Review the information in the Welcome dialog and click **Next**.
3. The License Agreement dialog is displayed. Review the terms of the license agreement and, if you agree to them, click **I accept the terms of the license agreement**. Then click **Next** to continue with the installation.

   The Installation Type dialog is displayed. By default, Typical will be installed.
4. Click **Next.** The Environment Type dialog is displayed.

   A TIBCO installation environment is used for software installations and consists of a Name and Directory. Products installed into different installation environments do not share components; therefore you can keep product installations completely isolated from each other.

   Since the GDSN component is an add-on component to MDM, ensure that you provide the MDM environment here. If you provide a different path, an error is displayed.
5. Click **Next**. A list of the components that will be installed, the location where they will be installed, and the total size is displayed.
6. Click **Install** for installation to proceed. The progress and the final results are displayed.
7. Click **Finish** to exit the wizard.

# GDSN Configuration Merging

After installing the GDSN plugin (over the MDM version), you need to merge the configuration. This is automatically done when you run the migration wizard, and this is the preferred way to migrate configuration.

You can only migrate MDM - MDM and GDSN - GDSN through the migration tool.

### Utility to Merge GDSN Configuration

If you need to merge configuration manually, use these utilities:

```
$MQ_HOME/datapool/bin/gdsnxmlPropMergeUtil.sh $MQ_HOME/datapool/bin/
gdsnxmlPropMergeUtil.bat
```

### Usage

```
<gdsnxmlPropMergeUtil> -GDSN -prop <MDM XML configuration file> -xmlin <GDSN
configuration XML file> -xmlout <Output configuration XML file>
```

# GDSN Specific Properties

The configuration properties are specific to GDSN.

*GDSN Specific Configuration Properties*

| Property | Configuration Outline | Description |
|---|---|---|
| Software Edition->Application Usage Profile | Basic | Defines whether the application is used for data synchronization (GDSN) or master data management (MDM) |
| Software Edition->Common Menus Configuration | Advanced | The configuration file location for common menus (navigation bar). |
| Integration Setup - External->Connector Rulebase | All | The rulebase to identify the protocol (for example, VELOSEL). |
| Rule Base->Synchronization Format Specific Attributes File | Advanced | Name of the Rulebase file used to customize format specific attributes on the synchronization profile View/Edit screen. |
| Rule Base->Send Message Screen Rulebase File | Advanced | The name of the rulebase file used to customize record attribute display on the SendMessage screen. |
| Rule Base->Work Item Description Rulebase File | Advanced | The name of the rulebase file to customize work item messages. |

## GDSN Predefined Components

For GDSN, a set of predefined components such as maps, rulebases, workflows, and templates are shipped with the application.

All these components are present in relevant folders under `$MQ_HOME/datapool/common/standard`

For details, see Standard Predefined Components.

# About TIBCO MDM Studio

TIBCO MDM Studio is a separate application, using which you can graphically design processes and repositories and later import in TIBCO MDM.

TIBCO MDM Studio comprises the following four components:

- Process Designer

  Processes once designed in the Process Designer, can be deployed directly in TIBCO MDM, without the need to manually export your process and import it into TIBCO MDM.

  Additionally, you can also import existing TIBCO MDM processes for modification and subsequent re-export. This is particularly useful for customers with existing processes which need to be modified or tweaked.

  TIBCO MDM currently supports processes defined as XML. TIBCO MDM Studio generates XPDL files that get converted to XML when you export. The XPDL is validated before being translated into native TIBCO MDM workflow format.
- Repository Designer

  The Repository Designer is based on TIBCO Business Studio and acts as an 'add on' component to Business Studio. The Repository Designer adds a visual element to designing repositories and makes the process quicker and more intuitive.
- Rulebase Designer

  The Rulebase Designer provides a graphical user interface for designing rules. The Rulebase Designer can be used to define rules from scratch or to edit existing MDM rules.
- UI Builder

  TIBCO MDM UI Builder is a new tool that has a cleaner, simpler, and more flexible UI in TIBCO MDM. TIBCO MDM UI Builder allows you to quickly, easily, and uniformly develop custom UIs by using simple drag and drop methods.

For more details, refer to the *TIBCO MDM Studio Process Designer*, *TIBCO MDM Studio Repository Designer*, *TIBCO MDM Studio Rulebase Designer,* and *TIBCO MDM Studio UI Builder* documentation.

# Configuration of TIBCO BusinessConnect and TIBCO BusinessWorks

You can configure TIBCO BusinessConnect for sending documents to and receiving documents from different data pools.

Refer to the appropriate documentation if you wish to use any other AS2 Gateway. TIBCO BusinessConnect 5.x supports JMS, however the TIBCO BusinessWorks plugin for BusinessConnect 5.x is still required.

### Prerequisites

- TIBCO BusinessConnect 5.3.0 must be installed on the system.
  Refer to the *TIBCO Business Connect documentation* for details.
- A database must be installed. BusinessConnect 5.x supports Oracle, MySQL and SQL Server database servers.
- TRA 5.6.1, Business Works 5.7.2, TIBCO Administrator 5.4.0, TIBCO EMS 5.1.2 must be installed on the system.
- UserID and password to access the TIBCO BusinessConnect Admin console.
- Port 4080 or any other port on which TIBCO BusinessConnect is listening needs to be opened.

## Setting up TIBCO Administrator

You need set up TIBCO Administrator to work with BusinessConnect.

### Procedure

1. Install TIBCO Administrator in console mode, read the Installation guide.
2. For Linux and UNIX setup, create Domain using domain Utility TIBCO_TRA_HOME/bin/domainutility.

   - Set defaults settings, give domain names as: *DOMAIN_NAME*/, choose UTF-8 for character set, default ports -> http port=9090, AJP 1.3 port =8009, shutdown port =8005, username=admin, password=admin
   - If UTF-8 doesn't work, change the characters to other character set
   - Change the owner to vsadmin and give all rights to tibco folder as -

     ```
     sudo chown -R vsadmin:vsadmin /home/tibco/
     sudo chmod -R 777 /home/tibco
     ```
3. On Windows, you can create a new domain using **Programs** > **TIBCO** > **TIBCO Runtime Agent & Adapter SDK 5.5** > **Domain Utility.**
4. After successfully configuring TIBCO Administrator, start TIBCO Administrator by executing the following on Linux or UNIX platform: *$TIBCO_HOME*/administrator/domain/*DOMAIN_NAME*/bin/tibcoadmin_*DOMAIN_NAME*/&
5. Start TIBCO Hawk Agent by executing the following on Linux / UNIX platform: *$TIBCO_HOME*/tra/domain/*DOMAIN_NAME*/hawkagent_*DOMAIN_NAME*/&
6. For Windows installations start TIBCO Hawk Agent must be started as NT service.
7. On Windows setup TIBCO Administrator must be started as NT service.
8. Access the administrator url as
   **http://*machinename*:9090/administrator/servlet/tibco_administrator**

## BusinessConnect Setup

You can configure BusinessConnect through a new configuration or import an existing setup from another installation and customize the imported BusinessConnect configuration.

### Prerequisites

- **Database and Database user**.
  A Database (which can be used by the BusinessConnect setup) should be configured.
- **TIBCO EMS**
  TIBCO EMS 5.1.2 should be installed and started as an NT service on Windows; on Linux it should started on root.
- **Install Unlimited Strength JCE Policy Files**
  To use BusinessConnect security features, download and install JavaTM Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files 1.5.0. To download and install the policy files, perform these steps:
  - Download the required files from the following web sites:
    - Windows and Linux: http://java.sun.com/j2se/1.5.0/download.jspIBM AIX and https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?source=jcesdk
  - Unzip `jce_policy-1_5_0.zip`
  - Copy `US_export_policy.jar` and `local_policy.jar` to:
    `$TIBCO_HOME\jre\1.5.0\lib\security`
- **TIBCO Administrator and TIBCO Hawk** TIBCO Administrator and TIBCO Hawk should be started.

## Configuring New BusinessConnect

Create a new BusinessConnect configuration.

### Procedure

1. Start TIBCO Administrator (**http://***machinename***:9090/administrator/servlet/tibco_administrator** ).
2. Login as **admin** for both user name and password.
3. Check if the BusinessConnect link appears in left side panel. If not, click **Application management** > **All service instance** > **machine name** > **Plug-Ins** > **Add** and browse to the bcwebadmin.war file of BusinessConnect.
   The WAR file can be found on:
   *`TIBCO_HOME`*/bc/5.3/admin-plugins/ bcwebadmin.war
   Adding the .war file takes some time.
4. On the left side panel, BusinessConnect option is available. Click **BusinessConnect** > **Manage Installation.**
5. In the Manage Installation window, specify the database details for installation and test connection for Oracle. The sample database details are:
   - JDBC Driver: tibcosoftwareinc.jdbc.oracle.OracleDriver
   - JDBC URL: jdbc:tibcosoftwareinc:oracle://*oracleservername*:1521;SID=*DATABASE_NAME*
6. Click **Test Connection** and **Done** if successful.

## Creating Participant of Type Partner

After creating a new BusinessConnect, create participant of type Partner.

### Procedure

1. Click **BusinessConnect** > **Participants** .
2. Select **Partner** in the **Participant** Type drop-down list and specify a name to the participant in the Participant Name field. Click **OK**.
3. In the **General** tab, specify all relevant details, select the **Active** check box to activate the participant.
4. In the **New Certificate** tab, specify the alias name and upload the certificate provided by the Trading partner (Data pool in case of TIBCO MDM). Click **Save**.
5. In the **Protocols** tab, click the **Enable** tab and select the **EZComm** protocol checkbox and click **OK**.
6. In the **Protocols** tab, click the newly enabled protocol EZComm.
7. In the Edit Enabled Protocol:EZComm window, click the **Add New** link to create a new AS2 ID.
8. In the New window, select the newly created AS2 ID and specify the new AS2 ID. Click **OK**.
9. In the Edit Enabled Protocol:EZComm window, click **Transports** > **Add** tab.
10. In the **New Transport** tab, select AS2_HTTP in the **Transport Type** drop-down list and specify the transport name. Click **OK**.
11. In the Edit AS2_HTTP Transport window, enter the details for the newly created transport. Change the URL as per the trading partner specification.
12. In the **Proxy Settings** tab, keep the default proxy settings as is.
13. Click **Save** to save the Participant.

## Creating Participant of type Host

After creating participant of type Partner, create a participant of type Host.

### Procedure

1. Click **BusinessConnect** > **Participants** .
2. Select **Host** in the **Participant** Type drop-down list and specify a name to the participant in the Participant Name field. Click **OK**.
3. In the **General** tab, specify all relevant details, select the **Active** check box to activate the participant.
4. Click CredentialsNew Private Key tabs. Specify the alias of the key, upload the private key (public key of which has sent to Partner). Click **Save**.
5. In the **Protocols** tab, click the **Enable** tab and select the **EZComm** protocol check box and click **OK**.
6. In the **Protocols** tab, click the newly enabled protocol EZComm **AS2 Identifier** > **Add new** link. In the AS2 Identity field, specify GLN on data pool. Click **Save** to save the Participant.

## Creating New Operations

After creating a participant of type Partner, create a participant of type Host.

### Procedure

1. Click **BusinessConnect** > **Operations Editor** .
2. In the Operations Editor window, click **Edit** for the EZComm protocol.

3. In the Edit Operations: EZComm window, click the **New Category** tab and select **EZComm** protocol and save the new category.
4. Click **New Version** to create a new version for new category. Specify details and **Save** the version.
5. Click the **New Operation** tab to create a new operation for this version.
6. In the Operation Type drop-down list, select **Asynchronous Request-Response** and click **OK**.
7. In the **Asynchronous Request-Response** tab, enter a name for the new operation.
8. In the **Request Action** tab and select the **Require Digital Signature** and **Require Content Encryption** check boxes.
9. Click the **Response Action** tab and select the check boxes as previously mentioned.
10. Click **Save** to save the newly created operation.

## Specifying System Settings

Specify the system settings for the newly created BusinessConnect.

### Procedure

1. Click the **BusinessConnect** > **System settings** .
2. Click **JDBC Configuration** in the right pane.
3. In the New JDBC Connection window, specify new JDBC connection details for BusinessConnect. Sample database details are:
    - JDBC Driver Class: tibcosoftwareinc.jdbc.oracle.OracleDriver
    - JDBC URL: jdbc:tibcosoftwareinc:oracle://localhost:1521;SID=bcdb<br>Database User: bcuser<br>Password: bcuser
4. Click **Test Connection**, if the connection is successful click **Save**.
5. In the Edit System Settings: JDBC Configuration window, select the newly created BusinessConnect connection in Audit Logs, Non Repudiation Logs, Runtime Data Store and click **Done**.

## Creating Business Agreement

Create a business agreement.

### Procedure

1. Click **BusinessConnect** > **Business Agreement** .
2. Click the **New** tab to create a new Agreement.
3. In the New Agreement window in the **Host party** and **Partner party** sections, select the names for which you want to make an agreement and click **OK**.
4. In the Business ConnectBusiness Agreements window in the **New** tab, click the newly created agreement.
5. In Edit Agreement window,
    - Select the **Valid** check box.
    - Select dates for the start and end of the agreement.
    - Click the **Add Protocol Binding** tab and select EZComm from available protocols to add a new protocol binding.

- Click the **EZComm** protocol link and specify the details.

6. In the Edit Protocol Binding: EZComm window, click the **Document Security** tab and select the following values:

- Signing Key: Name of the TIBCO's private certificate.
- Digest Algorithm: MD5
- Encryption Certificate: Name of Trading Partner Certificate
- Encryption Algorithm: Encryption algorithm used by Trading Partner
- Verification Certificate: Name of Trading Partner Certificate
- Decryption Key: Name of the TIBCO's private certificate

7. Click the **Transports** tab and select the following values:

- Primary Transport: 1SYNC_AS2
- AS2 MDN Async Reply Transport: 1SYNC_AS2
- AS2 Remote Server Certificate: 1Sync-PreProd-Certificate
- Client Authentication Identity for HTTPS, FTPS, HTTPSCA: TIBCO_private_key
- Client Authentication Identity for SSHFTP: None

8. Click **Save**.

## Deploying the BusinessConnect Configuration (Single Server Mode)

Deploy the newly created BusinessConnect configuration in a single server mode.

### Procedure

1. Deploy the configuration, click **Administrator** > **BusinessConnect** > **System Settings** > **Deployment Configuration** click **Create deployment Configuration**. The Business Connect link appears in Application Management.
2. Select a computer to deploy the BusinessConnect installation.
3. Click **Application Management** > **BusinessConnect** > **Configuration** > **Single Server.par.**
4. Select the computer name. If you do not see the computer name, it may be due to the hawk agent.
5. Select **Application Management** > **BusinessConnect** > **Configuration** > **BusinessConnect** > **Configuration** and click **Save**.
6. Click **Application Management** > **BusinessConnect** > **Configuration** > **Business Connect** > **HTTP** .
   a) In the Edit Transport Settings: HTTP, ensure the port is enabled and a proper value is provided.
7. Click **Application Management** > **BusinessConnect** > **Configuration** > **BusinessConnect** .
8. In the Edit Application Configuration: *computername* window, click the **Private Process Configuration** tab and select the **JMS** option.
   a) Click the **JMS** link and specify the following JMS details:

- JNDI Context Factory: com.tibco.tibjms.naming.TibjmsInitialContextFactory
- JNDI Context URL: tcp://localhost:7222
- Topic Connection Factory: TopicConnectionFactory
- Queue Connection Factory: QueueConnectionFactory

a) After completing all details click **Test connection** to test the JMS connection. Save the details, if the test connection is successful.

9. Click **Application Management** > **BusinessConnect** > **Configuration** > **BusinessConnect** > **Deploy** .
   a) Click the **Deploy** Tab. In the Deploy Configuration window, ensure that the **Start successfully deployed services** check box is selected.
   b) Check the status of BusinessConnect availability.
10. Click the **Application Management** > **BusinessConnect** > **Service Instance** . The BusinessConnect instance status is displayed as either stand by or running.
11. If the deployment BusinessConnect instance is not started, click **Application Management** > **BusinessConnect** > **Service Instance** > **BusinessConnect** instance check box and click **Start**.

    If every thing is successful, the BusinessConnect instance should be in running state. If the BusinessConnect instance cannot be started successfully, check the TIBCO Administrator logs (*TIBCO_HOME* administrator\domain\*DOMAIN_NAME*\logs\audit.txt) to find the cause.

## Setting up BusinessWorks

Set up BusinessWorks and import your project.

### Prerequisites

- Before you start a BW project, create a directory as follows: *TIBCO_HOME*\fsjndi\designerExtendedJars
- Copy the following JAR files inside the newly created directory:

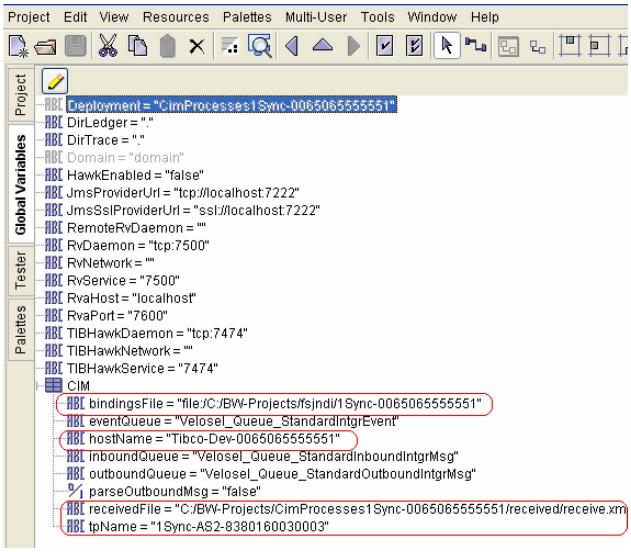
  -
    ```
    fscontext.jar
    jndi.jar
    providerutil.jar
    ```
- Open the designer.tra file located in *TIBCO_HOME*\designer\5.5\bin and add following entry: tibco.class.path.extended %CUSTOM_CP_EXT%%PSP%%STD_CP_EXT%:*TIBCO_HOME*/fsjndi/designerExtendedJars

### Procedure

1. Start TIBCO Designer.
2. In the **Project** tab, click **New empty project**.
3. In the Save Project window, select the project directory and TIBCO messaging encoding.
4. Click **Project** > **Import Full Project** .
5. Import vcrepo.dat from the existing BW project directory (directory of the same name as your existing BW configuration).
6. Alternately, if you do not want to import the existing project, copy the existing BW project directory with another name. This is similar to importing the existing BW project.
7. Edit the project as mentioned in the step 5 and 6.
8. Change the following global variables per the project requirement:

   - bindingFile: Provide the directory location where you want to place your binding file for this project.
   - receiveFile: Provide the directory location where you want received messages to be saved.
   - hostName: Provide the name of the Participant which you configured in BC as Participant of type Host.
   - tpName: Provide the name of the Participant which you configured in BC as Participant of type Partner.

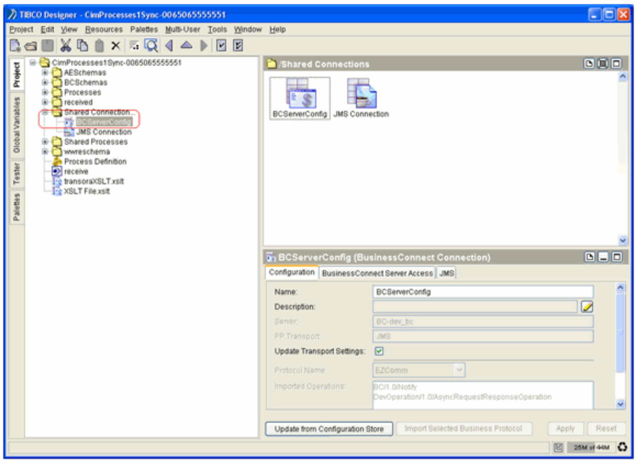
Sample values are as follows:


Project Edit View Resources Palettes Multi-User Tools Window Help
Project
Global Variables
Tester
Palettes

```
Deployment = "CimProcesses1Sync-0065065555551"
DirLedger = "."
DirTrace = "."
Domain = "domain"
HawkEnabled = "false"
JmsProviderUrl = "tcp://localhost:7222"
JmsSslProviderUrl = "ssl://localhost:7222"
RemoteRvDaemon = ""
RvDaemon = "tcp:7500"
RvNetwork = ""
RvService = "7500"
RvaHost = "localhost"
RvaPort = "7600"
TIBHawkDaemon = "tcp:7474"
TIBHawkNetwork = ""
TIBHawkService = "7474"
CIM
  bindingsFile = "file:/C:/BW-Projects/fsjndi/1Sync-0065065555551"
  eventQueue = "Velosel_Queue_StandardIntgrEvent"
  hostName = "Tibco-Dev-0065065555551"
  inboundQueue = "Velosel_Queue_StandardInboundIntgrMsg"
  outboundQueue = "Velosel_Queue_StandardOutboundIntgrMsg"
  parseOutboundMsg = "false"
  receivedFile = "C:/BW-Projects/CimProcesses1Sync-0065065555551/received/receive.xm
  tpName = "1Sync-AS2-8380160030003"
```

9. Click the **Project** tab**Shared Connection**. If you have already a BusinessConnect connection configured, update the connection.

10. Click **Update from Configuration Store**.



Name: BCServerConfig

Description:

Server: BC-dev_bc

PP Transport: JMS

Update Transport Settings: ☑

Protocol Name: EZComm

Imported Operations: BC/1.0/Notify
DevOperation/1.0/AsyncRequestResponseOperation

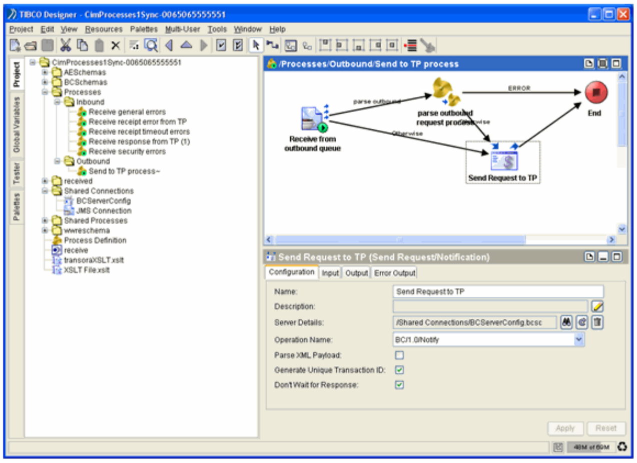
11. Select the **Select operations** check box; click **OK** on the next screen.

12. Click **Import selected Business Protocol**.

13. In the **BusinessConnect Server Access** tab, click **Update from Configuration Store** followed by **Import Selected Business Protocol**.

14. In the **JMS** tab of the previous screen, click **Update from Configuration Store** followed by **Import Selected Business Protocol**.

    If you have created a new BusinessConnect connection, ensure the property 'Server Details' of all the activities to a valid BC connection. A sample activity 'Send Request to TP' of process 'Send to TP Process' is as follows:



Name: Send Request to TP

Description:

Server Details: /Shared Connections/BCServerConfig.bcsc

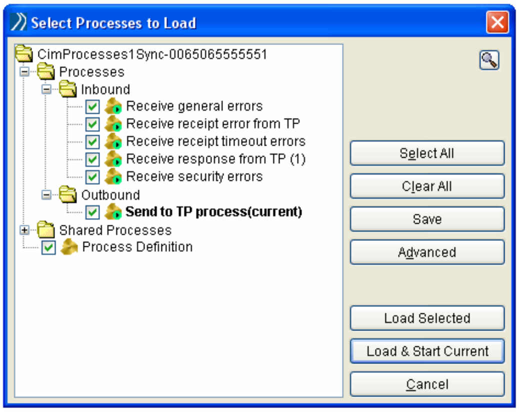
Operation Name: BC/1.0/Notify

Parse XML Payload: □

Generate Unique Transaction ID: ☑

Don't Wait for Response: ☑

15. Start the test engine by clicking the **Tester** tab on left side followed by green start signal.



☑ Receive general errors

☑ Receive receipt error from TP

☑ Receive receipt timeout errors

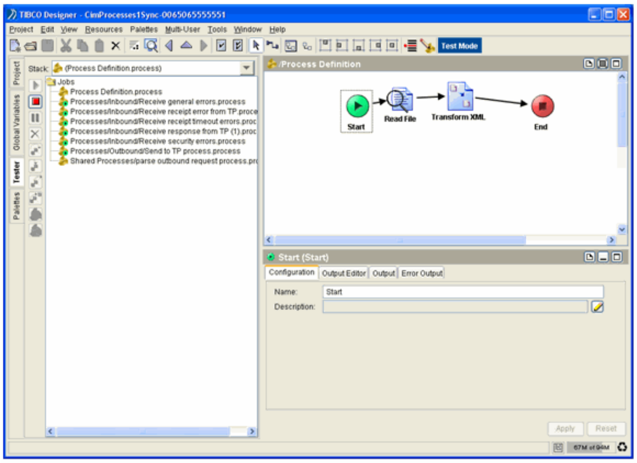
☑ Receive response from TP (1)

☑ Receive security errors

☑ **Send to TP process(current)**

☑ Process Definition

16. Click **Load selected**. After successful start, all processed should be visible.


TIBCO Designer - CimProcesses1Sync-0065065555551
Project Edit View Resources Palettes Multi-User Tools Window Help
Test Mode
Project
Global Variables
Tester
Palettes
Stack: (Process Definition.process)
Jobs
Process Definition.process
Processes/Inbound/Receive general errors.process
Processes/Inbound/Receive receipt error from TP.proce
Processes/Inbound/Receive receipt timeout errors.proc
Processes/Inbound/Receive response from TP (1).proc
Processes/Inbound/Receive security errors.process
Processes/Outbound/Send to TP process.process
Shared Processes/parse outbound request process.pr
/Process Definition
Start
Read File
Transform XML
End
Start (Start)
Configuration
Output Editor
Output
Error Output
Apply
Reset
67M of 94M

Name: Start

Description:

## Testing Connectivity of TIBCO MDM BusinessWorks BusinessConnect

After the entire configuration is done, you can test the connectivity of TIBCO MDM BusinessWorks BusinessConnect.

Prerequisites

- Have a ready to publish catalog in TIBCO MDM.

- In TIBCO Designer, ensure the BW tester is running, check for Processes\Outbound\Send To TP Process, and all setup done.
- The Binding file accessed by BW setup should have the IP of the machine from where testing has to be done.
- The BC engine should be running and all setups should be done.

### Procedure

1. Open TIBCO Administrator.
2. Publish product from TIBCO MDM.
3. Check BW for Processes\Outbound\Send To TP Process, whether the message received, as the activities flow shows green.
4. Check TIBCO Administrator for BC, in Log Viewer, for Audit Logs, for selected time range, whether the message reached and whether the message went out to the trading partner.
5. Check for acknowledgement from the datapool.
6. Check for response in BC, BW and TIBCO MDM eventLog.

   TIBCO BusinessConnect EZComm Protocol does not record advisory messages, such as the Responder Acknowledgement, in the audit log.

# Upgrade to TIBCO MDM

You can upgrade from a previous release of TIBCO MDM to the current release. Refer to the Release Notes for additional information.

Use the following options if you are migrating to release 9.2.0 from an earlier release.

- Use the Migration Wizard

  The Migration Wizard provides a easy to use interface for migration. Use the Migration Wizard for migrating from 8.3.2, 9.0.x, and 9.1.x versions of TIBCO MDM to 9.2.0. For details, see Migrating Using Migration Wizard.

- Use Individual Scripts

  The TIBCO MDM Migration utility internally invokes a number of scripts which perform migration of different components. These scripts are also directly available to you, if you need to migrate any component individually. For details, see Manually Migrating Individual Components.

## Premigration Steps

The premigration steps are common irrespective of how you choose to migrate.

### Procedure

1. Collect Data

   Ensure you have complete information to upgrade to the release of TIBCO MDM (Application server information, JMS information, Database information, and so on).

2. Stop and un-deploy the Existing Version of TIBCO MDM

   Stop the application and un-deploy TIBCO MDM from the application server. Do not run the previous uninstaller for TIBCO MDM installation.

3. Change Database Type and Copy JDBC Related Libraries
   Ensure that you change the database type in the Configurator (**admin** > **Settings** ) before migration. For example, if the database is SQL Server, change the database to SQL Server. Additionally, copy the JDBC driver JAR files required for each database into `$MQ_HOME/configurator/tomcat/lib`. For information on downloading or copying JAR files for each database, refer to the JDBC Related Libraries section in Third Party Libraries.

4. Backup TIBCO MDM data

   Backup the database schema,`COMMONDIR`, and application files (previous version of `$MQ_HOME`) that has the previous version of TIBCO MDM data. You can ignore or delete the `Temp` folder located at `COMMONDIR` before backing it up.

   Though not mandatory, TIBCO recommends this structure to have a consistent backup and binary deployment process of the product.

   - `tibcocim/rep`: refers to TIBCO MDM repository.
   - `tibcocim/backups`: stores TIBCO MDM backup.
   - `tibcocim/binaries`: stores copies of TIBCO MDM distributed binary JAR files.

5. Create Stable Data (Only for the Oracle database)

   If you have installed the Oracle database, you need to create the stable data.

   a) Navigate to `$MQ_HOME/db/oracle/migration/PrerequisiteFor83` and run **`CheckDataForConstraints.sql`** and **`CheckTableNames.sql`** files.

   **`CheckDataForConstraints.sql`**: displays the data that is not stable or includes the hanging pointers, such as the parent and child tables are related to each other. However, only the child table has reference to the parent table, and the parent table does not have reference records. In such case, remove or add the data in the parent table.

**`CheckTableNames.sql`**: displays TIBCO MDM reserved table names, constraint names, and indexes names if already exists. Rename the duplicate table names, and run the migration.

b) Navigate to `$MQ_HOME/db/oracle/migration/PrerequisiteFor83` and run the **`alterSequence.sql`** file.

**`alterSequence.sql`**: if the sequences are not migrated in the migration process or lost, run the **`alterSequence.sql`** file to restore all the necessary sequences.

6. Set Environment Variables

   a) The `$NODE_ID` variable must refer to the previous environment and the `$MQ_CONFIG_FILE` variable point to the current environment.

   b) The `$MQ_HOME` must point to the new environment.

7. Install Hotfix

   If you are migrating from the 8.3.2, 9.0.x, and 9.1.x versions, ensure that the previous version has the latest released hotfix installed.

8. Copying Optional files

   To migrate optional files from the `config` folder and `$MQ_HOME` of the previous installation to the new installation, set the properties to `true` in `$MQ_HOME/config/migrateConfigFolderFiles.prop`. For example,

   a) To copy `allmenu.xml` and `defaultdata.xml` files from the previous installation to new installation, set the file path and its value as follows: `config/security/allmenu.xml= true` and `config/security/defaultdata.xml= true`

   b) To copy `CronSchedules.xml` file from the previous installation to new installation, set the file path and its value as follows: `config/CronSchedules.xml=true`

   c) For TIBCO MDM Add-on for GDSN, by default the plug-in folder is copied from the previous installation to the new installation. If you do not want to copy the `plugins` folder, set the `plugins=false`.

9. Upgrade to Microsoft SQL Server 2016

   Upgrade to Microsoft SQL Server 2016, and then follow the database migration process.

## Migrating Using Migration Wizard

A Migration Wizard provides a graphical user interface to help users migrate from the previous versions, such as 8.3.2, 9.0.x, and 9.1.x versions of TIBCO MDM to the latest version.

Migration done through the wizard includes migration of rules, database, `ConfigValues.xml`, `FileWatcher.xml`, `CacheConfig.xml`, and other dependent files (such as workflows, maps, and rulebases).

### Procedure

1. Log in to Configurator.

2. Go to **Hi, [username]** > **Settings** .
   Change the database as per your requirement.

3. Go to **Tools** > **Migration Wizard** .

4. In the Specify Upgrade Details screen, specify the upgrade details. This screen handles version and edition to which migration should be done.

   a) The New Installation section displays the following fields:

   - **Location** (`MQ_HOME`): This is the installed path of your new version; your new `MQ_HOME` location is automatically detected.

On Windows operating systems, while specifying the path, you can use a backslash (\) as well as a forward slash (/).

- **Version**: The latest version to migrate to is also detected.
- **GDSN enabled**: By default, this is set to **No**, unless you have installed the GDSN plugin in which case it will be set to **Yes**. This is a read only value and cannot be changed.

a) The Previously Installed Version section displays the following fields:

- **Location** (`MQ_HOME`): Select the previously installed TIBCO MDM location.
- **Version**: This is the version you are migrating from. The following options are displayed in the drop-down list:
  - 8.3.2
  - 9.0.x : refers to 9.0.1
  - 9.1.x : refers to 9.1.0
- **GDSN enabled**: Select when your previous version was GDSN.

a) Click **Next**.

For the GDSN Edition, first you have to migrate MDM to MDM, install the GDSN latest version, and then run **Add-on Plug-in Installer**. You need to copy the plug-in directory from the previous `MQ_HOME` to new `MQ_HOME` directory.

5. In the Identify Location screen, specify existing and new common directory (`MQ_COMMON_DIR`) and configuration (`ConfigValues.xml`) information.

- New `MQ_COMMON_DIR` path (the latest version `MQ_COMMON_DIR` directory)
- New `ConfigValues.xml` path (the latest version `ConfigValues.xml` location)
- Existing `MQ_COMMON_DIR` path (previous `MQ_COMMON_DIR` directory)

  If the existing path of `MQ_COMMON_DIR` is set in the environment variable then it is populated or else by default, the new `MQ_COMMON_DIR` path is displayed. Manually enter the existing `MQ_COMMON_DIR` path.

- Existing `ConfigValues.xml` path (previous version `ConfigValues.xml` location)

  Click **Next.**

  Ensure that you have specified the correct paths for successful migration. In some cases, if you enter incorrect paths (for example, for `ConfigValues.xml`) during migration, the migration summary shows Successfully migrated message even when the migration is not successful and errors are seen in `$MQ_HOME/log/configValues.log`.

6. In the Select Migration Type screen, select the migration type. You can opt to do a complete migration or a custom one where you select the components.

a) **Complete Migration**: If you choose to do a complete migration, all components are migrated, except rules which need to be migrated separately (this is because the TIBCO MDM server needs to be up before migrating rules).

b) **Custom Migration**: In a custom migration, select from the following components:

- Database
- `ConfigValues.xml`
- Dependent files (migration of workflows, `FileWatcher.xml`, rules, and so on.)
- GDSN Add-on files (if applicable)

c) **Rules Migration**: If you choose rules migration, custom process are migrated, default template and new process selection rules are added.

> - Before running the rules migration, you need to perform the prerequisites. For information, refer to the section, "Rules Migration" in Manually Migrating Individual Components.
> - For rules migration, ensure that TIBCO MDM server is up and running.
> - Previous `MQ_COMMON_DIR` is updated during migration to the latest version. You must use the latest version `MQ_COMMON_DIR`.
> - New `ConfigValues.xml` is updated during migration to latest version. You must use the latest version `ConfigValues.xml`.

7. In the Database Migration screen, specify the following details of the existing database:

> This dialog is only displayed if you have selected the Complete Migration option or if you have selected Database as a component under Custom Migration option in Step 5.

- **Database Host**: Specify the database host name or IP address.
- **Database Port** : Specify the database port.
- **Database Name**: The database name.
- **User Name**: The user name for the database.
- **Password**: The password for the database.
- **MDM Tablespace**: Name of the table space. For example, **VELODBDATA1**. This field is enabled only if you have selected the Oracle database type.

8. Click **Finish** to start the migration.

   If any one of the following components failed to migrate, an error message is displayed:

   - Database
   - `ConfigValues.xml`
   - Dependent files
   - GDSN add-on files

9. In the Error window,

   a) Click **Open** to view the logs.
   b) Click **Abort** to stop the migration.
   c) Click **Ignore** to continue the migration.

> You can ignore errors, such as dropping non-existing tables, views, triggers, and so on; displayed during database migration. Such errors are displayed in the database migration log located at `$MQ_HOME/db/`*`database_type`*`/migration/databaseMigration.log`.For example,
>
> ```
> ERROR:-Cannot drop the index
> 'MV_SHARED_BOOLEAN_TABLE.MV_SHARED_BOOLEAN_TABLE_IDX', because it does
> not exist or you do not have permission.
> ```

### Result

A summary of the migration is displayed along with output logs, which can be downloaded.

## Manually Migrating Individual Components

Individual scripts for each component migration are also available in the `$MQ_HOME/bin/migration` folder. If you want to migrate an individual component, you can run each script separately.

| Component to Migrate | Script Name and Location | Description | Usage |
|---|---|---|---|
| Database Migration | `$MQ_HOME/bin/migration/`<br>`DatabaseMigration.bat/sh` | Use this utility to migrate the database from the previous version to the new version. | `DatabaseMigration -path DBProviderName DatabaseName Username Password Server Name`<br>Where *path* can be: -83_92, -90_92, or -91_92<br>Where *DBProviderName* can be: ORACLE or SQLSERVER |

| Component to Migrate | Script Name and Location | Description | Usage |
|---|---|---|---|
| Configuration Migration | • For MDM: `$MQ_HOME/bin/migration/ConfigValuesMergeUtil.sh/bat`<br>• For GDSN: `$MQ_HOME/plugins/gdsn/`*`version_number`*`/bin/gdsnxmlPropMergeUtil.bat` | Use this utility to migrate `ConfigValues.xml` from the previous version to the new version. | `<ConfigValuesMergeUtil> -`*`path`*` -prop <`*`version`*` XML configuration file> -xmlin <Source configuration XML file> -xmlout <Output configuration XML file>`<br>Where,<br>• *path* can be: -83_92, -90_92, or -91_92<br>• `<`*`version`*` XML configuration file>` is the previous version of `ConfigValues.xml` with an absolute path<br>• `<Source configuration XML file>` is the new version of `ConfigValues.xml`<br>• `<Output configuration XML file>` is the absolute path to an user `ConfigValues.xml` file. For example, `$MQ_HOME/config/Migration_Config_Values.xml`.<br>After migration, back up the new `ConfigValues.xml` file, and then rename the `Migration_Config_Values.xml` file to `ConfigValues.xml` to use it. |
| | | | (when merging MDM and GDSN Configuration)<br>`<gdsnxmlPropMergeUtil> -GDSN -prop <9.1 MDM XML configuration file> -xmlin < 9.1 GDSN configuration XML file> -xmlout <Output configuration XML file>` |

| Component to Migrate | Script Name and Location | Description | Usage |
|---|---|---|---|
| Configuration Directory Migration | • Indexer Configuration: `$MQ_HOME/bin/migration/CopyConfigDirFiles.bat/sh`.<br>• Cache Configuration: `$MQ_HOME/bin/migration/CacheConfigMergeUtil.bat/sh`. | • The Indexer configuration refers to the `IndexerConfig.xml` file. This is an optional file, migrate if needed.<br>The utility copies specific files from the old `Config` directory to the new `Config` directory based on `migrateConfigFolderFiles.prop` configuration.<br>• The utility upgrades the `CacheConfig.xml` file in the `Config` directory. The `CacheConfig.xml` file includes setup and configuration of ActiveSpaces cache. | `<CopyConfigDirFiles.bat> -<path>`<br>Where *path* can be: -83_92, -90_92, or -91_92<br>`<CacheConfigMergeUtil> -<path> -prop <version CacheConfig file> -xmlin <Source CacheConfig XML file> -xmlout <Output CacheConfig XML file>"`<br>Where,<br>• <*path*> can be: -83_92, -90_92, or -91_92<br>• `<versionCacheConfig file>` is the previous version of `CacheConfig.xml` with an absolute path<br>• `<Source CacheConfig XMLfile>` is the new version of `CacheConfig.xml`<br>• `<Output CacheConfig XML file>` is the absolute path to an user `CacheConfig.xml` file. |

| Component to Migrate | Script Name and Location | Description | Usage |
|---|---|---|---|
| Rules Migration | `$MQ_HOME/bin/migration/ MigrateRules.bat./sh` | This utility adds new rules and new default template to the existing organization, and it adds the new process selection rule. You can also migrate custom processes, forms, maps, and rulebases.<br>**Prerequisites:**<br>• Ensure that the TIBCO MDM server is up before running this utility. Also ensure that the *OLD_MQ_HOME* variable is set.<br>• Ensure that you have provided organization ID in %MQ_HOME%`\bin\migration\orglist.txt` and `orglist-processSelection.txt`.<br>• The organization IDs are procured from the ID column of the `ORGANIZATION` table. | The following parameters are mandatory:<br>• `-addNewProcessSelectionRules`: Adds new process selection rule to the existing organization as defined in `orglist-processSelection.txt`.<br>• `-addDefaultTemplate`: Adds new default template existing rule in organization as defined in `orglist-processSelection.txt`.<br>• `-addNewRules`: Adds new rules to the existing organization as defined in `org.list`.<br>• `-migrateRulesActions`: Migrates rules. |

| Component to Migrate | Script Name and Location | Description | Usage |
|---|---|---|---|
| FileWatcher Migration | `$MQ_HOME/bin/migration/fileWatcherMergeUtil.bat/sh` | This utility allows you to migrate the `FileWatcher.xml` file from the previous version to the new version. | `fileWatcherMergeUtil -<path> -prop <old FileWatcher file>. -xmlin <new FileWatcher file> -xmlout <Output FileWatcher file>`<br>Where <*path*> can be:<br>-83_92, -90_92, or -91_92 |
| Common Directory Migration | `$MQ_HOME/bin/migration/CopyCommonDirFiles.bat/sh` | This script replaces standard files from 8.x versions to 9.1. | `CopyCommonDirFiles -<path>`<br>Where <*path*> can be:<br>-83_92, -90_92, or -91_92<br>- It prompts for new `$MQ_COMMON_DIR` (Current version `$MQ_HOME/common`)<br>- It prompts for previous `$MQ_COMMON_DIR` (Previous version `$MQ_HOME/common`)<br>**Note**: Previous `$MQ_COMMON_DIR` is updated during migration to 9.1 version. You must use this `$MQ_COMMON_DIR` |

| Component to Migrate | Script Name and Location | Description | Usage |
|---|---|---|---|
| Golden Bundle Migration [Optional] | `$MQ_HOME/db/`*`databasename`*`/migration/Migrate83_90/fixRelationshipGoldenBundle.sql` | Using this script, you can use the golden bundle feature for the existing records. To enable the golden bundle feature, set the value of the `Record UI golden view` configuration property to `true` in Configurator. The property enables link to view golden version of records and bundle. By default, the value is set to `false`.<br>The script ensures that the relationship between the golden copies is established and also the relationship between the latest version and the golden copy version. | Copy the `fixRelationshipGoldenBundle.sql` script to any database tool and execute as a script. Or<br>If the database client is installed, navigate to the `$MQ_HOME/db/database/migration/Migrate901_91` and connect to the database, and then run the `fixRelationshipGoldenBundle.sql` script using the appropriate command based on the database. For example: **`Oracle@fixRelationshipGoldenBundle.sql`** |

| Component to Migrate | Script Name and Location | Description | Usage |
|---|---|---|---|
| Synchronization Migration [Optional] | `$MQ_HOME/db/`*`databasename`*`/migration/Migrate83_90/alterForSelfOutputMaps.sql`<br>Run the `alterForSelfOutputMaps.sql script` only if you are migrating from the 9.0.x version or previous version. | Run the script for the following:<br>• to support out-of-the-box self relationships between output maps for the existing metadata<br>• to add self related output maps to the repository that contains self active relationships because the relationships mentioned in the synchronization workflow are no more supported. | Execute the script on the latest version database because by default the script works for all enterprises and repositories. You can execute the script on a particular enterprise by uncommenting the following condition in the `WHERE` clause and providing an organization ID:<br>Remove -- to uncomment the condition and replace organization ID by removing #.<br>`WHERE c.type = 'OUTPUTCATALOG' -- AND c.sourceorganizationid = #` |
| Work Item Migration | `$MQ_HOME/db/`*`databasename`*`/migration/Migrate901_91/createAndMigrateWIAttributes.sql`<br>Run the `createAndMigrateWIAttributes.sql` script only if you are migrating from the 9.0.x version or previous version. | Run the script to migrate the values of work item attributes from the `WorkItemDetail` table to the `WorkItemAttribute` table. When the value of an attribute is changed through work item modification, the value in the `WorkItemAttribute` table is updated. The table is used for sorting of work items based on configured attributes. | Execute the script on a particular enterprise. |

> ⚠ Ensure that you run all utilities from the directories in which they are present (such as `$MQ_HOME/bin/migration`); do not run it from remote locations by providing the absolute paths.
>
> For example, do not run a script from a remote directory by providing `$MQ_HOME/bin/migration/`*`scriptname`*`.sh`. Instead navigate to the `$MQ_HOME/bin/migration` directory and then run *`scriptname`*`.sh`.

## Postmigration Steps

- Copy the `hibernate3.jar` file from `$MQ_HOME\lib\external\hibernate` to `$MQ_HOME\lib\external` of the latest version.
- On migration, all the old workflows at the enterprise level are migrated.
- If any standard workflows exist with the custom activity, the old workflows are backed up and replaced with new workflows on migration.
- Custom workflows are not migrated. Previously, if you have customized any standard workflows, after migration you need re-customize them.
- For the existing users, the old password is applicable until it expires. To enforce the new password policy, keep the `Password expiry period` = `0`, and change `Forgot Password` to `true`. For information on password configuration properties, refer to the section, "Configuration Properties for Password" in *TIBCO MDM System Administration*.
- To resolve the migration errors that you may come across postmigration, refer to the section Migration Troubleshooting.

## Limitation

For TIBCO MDM, if Microsoft SQL Server is on Linux platform, migration to 9.2.0 is not supported.

# Applying Hotfixes

An updated `customUtil.sh` file is provided with a hotfix. Extract it to the `$MQ_HOME/build/custom` directory.

### Procedure

1. Uninstall TIBCO MDM from the Application Server.
2. Create the directory `$MQ_HOME/customEAR`.
3. Copy the hotfix TAR file to `$MQ_HOME/customEAR`.
4. Change directory to `$MQ_HOME/customEAR` and untar the hotfix tar file.
5. Copy `customUtil.sh` to `$MQ_HOME/build/custom`.
6. Change directory to `$MQ_HOME/build/custom` and run the following command:
   ```
   ./customUtil.sh -updateEarFile
   ```
7. Enter **y** when prompted with "Ready to continue".
8. Enter the name of the hotfix JAR file when prompted for.

   This creates an updated EAR file in the `$MQ_HOME/customEAR` directory.
9. Install the updated `ECM.ear` file in the Application Server.

   > Refer to the *TIBCO MDM Readme* for more details.

## Rolling Upgrades and High Availability Configuration

You can upgrade application servers using rolling upgrade, however with a few conditions.

The following conditions must be considered:

- If database is changed, everything must be upgraded.
- If application servers share configuration, upgrade requires change to configuration.

### Version of Infrastructure

In general, most infrastructure upgrades require upgradation of all the components at the same time.

- Database version – All database servers must be upgraded at the same time.
- Operating system upgrade – Most operating system version upgrades can be done one server at a time depending on compatibility matrix published by TIBCO and OS vendor. In fact, all the servers are not required to be on the same OS platforms.
- Web server upgrade – Web server upgrade can be done independent of other upgrades, each web server can be upgraded.
- JMS server – JMS server can be upgraded independent of other servers. Within a JMS cluster, each of them must be upgraded at the same time.

### TIBCO MDM Application Sever Upgrade

- Application server version upgrade – As long as application server version is supported by a TIBCO MDM release, it can be upgraded one at a time.
- TIBCO MDM version upgrade – TIBCO MDM version may require to upgrade.
- Database schema or seed data changes – If schema changes are required, all TIBCO MDM instances must be upgraded together.
- Configuration file changes – Each server can be upgraded by using a new `ConfigValues.xml` file; while the previous version continues to use the previous config file.
- Executables – In some cases, all servers need to be upgraded at the same time (assuming there are no database schema changes). For example, when an object distributed over queue has changed, it requires that all recipient are on the same version to avoid de-serialization errors.
- Cache server – The Cache server may require a restart when TIBCO MDM server is upgraded due to change in data objects, which are stored in cache. In this case, all TIBCO MDM instances and all cache instances must be upgraded at the same time.
- Advanced Search Engine – It is shipped with TIBCO MDM and follows the same upgrade path as TIBCO MDM version upgrade.

### High Availability

- Each component can be clustered
  - Database, that is, Oracle RAC
  - TIBCO MDM instances
  - Web servers
  - Advanced Search Engines (Patterns)
  - Cache servers
  - JMS servers

- TIBCO MDM can be configured to use clusters of other engines (database, cache, JMS, Advanced search engine, and so on).
- When a component fails, work is transferred to another server except for the following points:
  - TIBCO MDM user sessions are not replicated. When TIBCO MDM server fails, in-progress user transactions are discarded and user is redirected to another server. The transactions or operations, which are incomplete only those are discarded. User must login again.
  - When workflows failover to another server; depending on the workflow configurations, sometimes a workflow activity may repeat. For example, when a work item is created, distributed cache is updated to indicate that the work item has been created. When workflow restarts, it does not generate duplicate work item if such a marker is found. However, if cache has also failed, this marker may be lost and a duplicate work item is generated. Same scenario applies for any outgoing messages generated by the workflow.
- The workload is shared amongst all engines.
  - TIBCO MDM instances share the workload using JMS queues. On failure of an instance, workload is automatically redistributed.
  - Cache can be setup to replicate data to more than one instance. On cache failure, critical cached data is transferred to another server or a replicated copy is used. Most of the cached data does not have to be replicated as it is persisted to database.
  - TIBCO MDM server automatically connects to the next database, Advanced Search Engine, cache, or JMS server.
- TIBCO MDM implements a *wait and retry* algorithm for transient system failures while executing workflows. For example, if an intermittent network failure happens, which causes database connection to be dropped, TIBCO MDM rolls back to the last commit state and retry the operation.
- The web server can be setup to automatically redirect the users to next working TIBCO MDM instance.

## Migration Troubleshooting

Resolve the migration errors that you may come across after migrating to TIBCO MDM.

| Issue | Description | Solution |
|---|---|---|
| TIBCO MDM server start-up failure after migration | After migration from the previous version to 9.0 version, the following exception may occur at the server startup:<br>`INF-7508: Total of all limits for distributed and near caches should not exceed 100 (percent).` | Manually update the `CacheConfig.xml` file. Perform any one of the following changes:<br>• Adjust limits of the `DISTRIBUTED` and `NEAR` cache types. Ensure the sum of these limits should be less than or is equal to (<=) 100.<br>• Replace the `<Limit>` cache attribute with the `<ListSize>` cache attribute. Specify the appropriate value for the `<ListSize>` cache attribute. |

| Issue | Description | Solution |
|---|---|---|
| TIBCO MDM server start-up failure after WildFly Application Server migration | When you migrate from the WildFly Application Server 14.0.0 to 16.0.0 version, the `javax.ejb.NoSuchEJBException: EJBCLIENT000079` exception occurs in the `error.log` file for loading of the FileWatcher and PreloadManager classes. | 1. Log in to Configurator.<br>2. Remove the following values from the existing configuration property [`Static Receiver Initialization List (com.tibco.cim.initialize.receiver.static)`] and add them to the new configuration property [`Static Receiver Lazy Initialization List (com.tibco.cim.initialize.receiver.static.lazy)`]:<br>• FileWatcher:`com.tibco.cim.init.FileWatcher`<br>• PreloadManager: `com.tibco.cim.init.PreLoadManager`<br>Moving these values to the `Static Receiver Lazy Initialization List` configuration property ensures lazy loading of the PreloadManager and FileWatcher classes, that is, the classes are loaded after starting the WildFly Application Server. |

# Standard Predefined Components

All standard predefined components are shipped with the application, which include standard MDM specific components and incremental GDSN specific components, if you have installed the GDSN plugin.

- MDM Components: includes maps, rulebases, workflows, templates, forms, samples, and misc. All these components are located in the relevant folders under `$MQ_HOME/common/standard`.
- GDSN Components: includes maps, rulebases, workflows, and templates. All these components are located in the relevant folders under `$MQ_HOME/datapool/common/standard`.

The out-of-the-box templates and samples of TIBCO MDM are located in the `$MQ_HOME/Standard` folder.

⚠ TIBCO does not recommend you to modify all components supplied in this directory. If a modification is required, maintain the modified file in an enterprise- specific directory.

## MDM Specific Maps

The translation maps are located in the `$MQ_HOME/common/standard/maps` folder.

*MDM Specific Maps*

| File | Description |
|---|---|
| `mp26importto26cateditadd.xsl` | This XSLT is used to translate a process definition from the xpdl2 format to the TIBCO MDM process definition (workflow) format. |
| `mp26importto26cateditchange.xsl` | This file translates the import workflow request mlXML document to a 'Record Modify' workflow request mlXML document and sets the document type and subtype before spawning the modify record subflow |
| `mp26importto26cateditdelete.xsl` | This file translates the import workflow request mlXML document to a 'Record Delete' document and sets the document type and subtype before spawning the delete record subflow. |
| `mpfrom26topdfprodspecv1.xsl` | This file provides a sample translation map for conversion of record data to PDF. The record data is in the mlXML format, typically output of one of the workflow activities. |
| `mpfromebxml21envelopetomlxml_Sample1.xsl` | This file removes the ebXML wrapper and extracts the mlXML payload from the received message. |
| `mpfromebxml21envelopetomlxml_Sample2.xsl` | This file removes the ebXML wrapper and extracts the mlXML payload from the received message. |

| File | Description |
|---|---|
| `mpfromebxml21envelopetounknown.xsl` | This file is used during CommStandardInboundIntgrMsg unmarshalling for converting an ebXML message into the mlXML format. Use this file if the ebXML payload is within CDATA in the envelope. |
| `mpfromebxml21envelopetounknownxml.xsl` | This file is used during CommStandardInboundIntgrMsg unmarshalling for converting an ebXML message into the mlXML format. Use this file if the ebXML payload is XML and is NOT within CDATA in ebXML envelope. |
| `mpfromebxml21totransportevent.xsl` | This file is used during the unmarshalling process for converting an ebXML message into internal commEvent format. |
| `mpfromunknowntoebxml21envelope.xsl` | This file is used by the CommStandardOutboundIntgrMsg marshalling pipeline to convert an internal message to the ebXML format before sending it to an external application. |
| `mptrcommandtype.xslt` | This file is used by the ProcessServiceMessage activity to translate the command type from Query to Update in the response XML before adding it to the response jar file. |
| `Native80ToNew80MetaDataConverter.xsl` | During export of metadata (when only repository export is requested), this stylesheet is used to convert application generated metadata into new format which is more structured, has a schema defined and hence offers better validation. |
| `New80ToNative80MetaDataConverter.xsl` | During import of metadata (when only repository import is being processed) this stylesheet is used to convert incoming metadata<br>(new format) into the native format which the application s and processes internally. Before this is done, incoming meta data is validated against a schema. |
| `xpdl2Tocim.xsl` | This XSLT is used to translate a process definition from the xpdl2 format to the TIBCO MDM process definition (workflow) format. |

## GDSN Specific Maps

The translation maps specific to GDSN are located in the `$MQ_HOME`/datapool/common/standard/maps folder.

*GDSN Specific Maps*

| File | Description |
|---|---|
| `mpfrom1sync62cicwlto26v1.xsl` | This XSLT translates an incoming CIC message (1SYNC Item Authorization Response) to the mlXML format. |
| `mpfromagentrics50cdnto26v1.xsl` | This XSLT translates an Agentrics CDN (Catalogue Delivery Notification) message to the mlXML format. |
| `mpfromagentrics50cicwlto26v1.xsl` | This XSLT translates an Agentrics CIC (Catalogue Item Confirmation) message to the mlXML format. |
| `mpfromagentrics50rfcinwlto26v1.xsl` | This XSLT translates an Agentrics RFCIN (Request for CIN) message to the mlXML format. |
| `mpfromagentrics50wlto26v1.xsl` | This XSLT translates an Agentrics CIN (Catalogue Item Notification) message to the mXML format. |
| `mpfromcatact26to1sync62v1.xsl` | This XSLT translates an outgoing mlXML message to the 1Sync Catalog Request message format (1SYNC Item Maintenance message). |
| `mpfromcatact26toagentrics50cicv1.xsl` | This XSLT translates an outgoing mlXML message to Agentrics CIC (Catalogue Item Confirmation) message format. |
| `mpfromcatact26toagentrics50rfcinv1.xsl` | This XSLT translates an outgoing mlXML message to the Agentrics RFCIN (Request for CIN) message. |
| `mpfromcatact26toagentrics50v1.xsl` | This XSLT translates an outgoing mlXML message to Agentrics CIN (Catalogue Item Notification) message format. |
| `mpfromcatact26tononebxml70v1.xsl` | This sample map file generates a non-ebxml outbound message. |
| `mpfromcatact26toveloselcinv1.xsl` | This file translates outgoing mlXML messages to a message format used in CIM-to-CIM synchronization. |

| File | Description |
|---|---|
| `mpfromcatact26toveloselcinv1_all_Attributes.xsl` | This file translates outgoing mlXML messages to the message format used in CIM-to-CIM synchronization. It maps all the catalog attributes rather than filtering out EAN.UCC format specific attributes. |
| `mpfromveloselcintoveloselcinrv1.xsl` | This file is used in CIM-to-CIM synchronization to generate a response for an incoming message. |
| `mpfromveloseltomlxml26v1.xsl` | This file translates an incoming message to the mlXML format during CIM-to-CIM synchronization. |

## MDM Specific Rulebases

The rulebases are located in the `$MQ_HOME/common/standard/rulebase` folder.

*MDM Specific Rulebases*

| File | Description |
|---|---|
| `allowAttrList.xml` | Use to specify the allow merge attributes. |
| `DisplayAttrList.xml` | Use to specify the name of the attributes that you want to display in the report. |
| `skipAttrList.xml` | Use to specify the skip merge attributes. |
| `MatchAttrList.xml` | Use to specify the name of matching attributes and weightage. Using the matching attributes, you can search for close, not necessarily exact, matching records. |
| `rbconnectorprotocolderivation.xml` | This rulebase decides the protocol to be used by messaging handler for outgoing messages or the response handler for incoming messages. |
| `rbdefault.xml` | This is the default synchronization choreography derivation rulebase. It provides a simple default operation for outgoing messages. |
| `rbexpirydate.xml` | This is a sample rulebase to demonstrate workitem expiry date calculation based on record attributes. This rulebase is used to determine if workitem is timed out if the expiry method is set to COMPUTE. |
| `rbformatattributes.xml` | This is a sample rulebase to customize synchronization profile screen. |

| File | Description |
|---|---|
| `rbgenerateid.xml` | This is an empty placeholder rulebase which is used in incoming message processing workflows. Customize this rulebase to provide a product ID generation algorithm, if needed. |
| `rbmessageprocessorderivation.xml` | This rulebase evaluates the message processor property key prefix(es) based on various parameters associated with the message being received. This prefix string is used to pick up the message processor class name from the Configurator to be instantiated to process the received message. |
| `rbpasswdval.xml` | This sample rulebase allows you to apply certain policies for valid passwords. |
| `rbresponsehandlerprotocolderivation.xml` | This rulebase is used to identify the protocol used for message processing. |
| `rbsendmessage.xml` | The generic screen is available from the Product View screen, where there is an actions link for SendMessage based on the user privileges (Role). The UI that opens on clicking this link is completely customizable based on the underlined rulebase `rbsendMessage.xml`. Currently, it is used to send an RFCIN message and generate fact sheet.<br>Following customizations are possible:<br>• Customize the product attributes that can be displayed.<br>• Customize action that can be performed.<br>• Customize the MarketPlace and TradingPartner Credentials.<br>• Customize the user defined attributes |
| `rbveloselsupplier.xml` | This rulebase identifies different synch operations for CIM2CIM. |
| `rbworkitemdescription.xml` | This rulebase derives out-of-box work-item descriptions. |
| `RulebaseCustomFunction.JAVA` | This is a sample rulebase custom function implementation. |
| `RulebaseCustomFunction.class` | This is a sample rulebase custom function implementation. |

| File | Description |
|---|---|
| `statusderivation.xml` | This rulebase identifies the response status received from the datapool and translates the status into a TIBCO MDM-specific status.<br>This rulebase is used to summarize synchronization status and to map the status to business terms. Default implementation is provided to support standard synchronization for Agentrics and 1Sync. |

## GDSN Specific Rulebases

The rulebases specific to GDSN are located in the `$MQ_HOME`/datapool/common/standard/rulebase folder.

*GDSN Specific Rulebases*

| File | Description |
|---|---|
| `_1sync_catalogvalidation.xml` | This rulebase declares validation rules for out-of-the-box catalog synchronization for 1Sync datapool. |
| `_1sync_mastercatalog_catalogvalidation.xml` | This rulebase declares validation rules for master catalog used in synchronization having 1SYNC catalog output map. |
| `_wwre_catalogvalidation.xml` | This rulebase declares validation rules for out-of-the-box catalog synchronization for WWRE (Agentrics) datapool. |
| `_wwre_mastercatalog_catalogvalidation.xml` | This rulebase declares validation rules for master catalog having WWRE (Agentrics) catalog output map. |
| `rbapprover.xml` | This is an empty rulebase which is used in the workflow `wfin26prodnotifretailerv4.xml` to flag any errors or warnings in the incoming message data. This rulebase should be customized as per your business process needs. |
| `rbconnectorprotocolderivation_gdsn.xml` | This rulebase decides the protocol to be used by messaging handler for outgoing messages or the response handler for incoming messages. |
| `rbconnectorrolederivation.xml` | This rulebase decides the protocol to be used by messaging handler for outgoing messages or the response handler for incoming messages. |
| `rbduplicate.xml` | This rulebase provides a sample duplicate check constraint for incoming messages. The sample rulebase is used to implement duplicate check for GDSN incoming messages. |

| File | Description |
|---|---|
| `rbeditor.xml` | This is an empty rulebase which is used in workflow `wfin26prodnotifretailerv4.xml` and other incoming message processing workflows to flag any errors or warnings in the incoming message data. This rulebase should be customized as per your business process needs. |
| `rbformatattributes_gdsn.xml` | Format specific attributes to be used for a marketplace. |
| `rbgdsnrelationship.xml` | This rulebase is used to identify the relationship to be used in the GDSN edition. |
| `rbGPCClassificationscheme.xml` | This rulebase drives the classification codes for the GPC predefined classification scheme. |
| `rbresponsehandlerprotocolderivation_gdsn.xml` | This rulebase is used to identify the protocol used for message processing. |
| `rbretailerval.xml` | This is a sample retailer validation rulebase. |
| `rbsendmessage_gdsn.xml` | The generic screen is available from the Product View screen, where there is an actions link for SendMessage based on the user privileges (Role). The UI that opens on clicking this link is completely customizable based on the underlined rulebase `rbsendMessage.xml`. Currently, it is used to send an RFCIN message and generate fact sheet.<br>Following customizations are possible:<br>• Customize the product attributes that can be displayed.<br>• Customize action that can be performed.<br>• Customize the MarketPlace and TradingPartner Credentials.<br>• Customize the user defined attributes |
| `rbsyncstatusupdatederivation.xml` | This sample rulebase is used to customize manual synchronization status update action. |
| `rbtolerance.xml` | Permissible values for attributes are defined in this rulebase. |
| `rbtransorasupplier.xml` | This rulebase identifies different sync operations for the 1SYNC datapool. |
| `rbUDEXClassificationscheme.xml` | This rulebase derives the classification codes for the UDEX predefined classification scheme. |

| File | Description |
|---|---|
| `rbworkitemdescription_gdsn.xml` | This rulebase derives out-of-box work-item descriptions for the GDSN edition. |
| `rbwwreretailer.xml` | This rulebase identifies different sync operations for the Agentrics (WWRE) datapool on the retailer side. |
| `rbwwresupplier.xml` | This rulebase identifies different sync operations for the Agentrics (WWRE) datapool on the supplier side. |

## MDM Specific Templates

The templates specific to MDM are located in the `$MQ_HOME/common/standard/template` folder.

*MDM Specific Templates*

| File | Description |
|---|---|
| `tm26catimportv1.xml` | Workflow request template for initiating import subflows for approval and conflict resolution. |
| `tm26catpubwcatv1.xml` | Workflow request template for initiating a synchronization workflow – typically used when a synchronization profile is used. Synchronization is initiated for all the records specified in the synchronization profile. |
| `tm26catpubwoutcatv1.xml` | Workflow request template for initiating a synchronization workflow – typically used when synchronization profile is NOT used. Synchronization is initiated for one bundle of records. |

## MDM Specific Workflows

The workflows specific to MDM are located in the `$MQ_COMMON_DIR/Standard/workflow` folder.

*MDM Specific Workflows*

| File | Description |
|---|---|
| `cancelworkflowv1.xml` | Predefined workflow to cancel event. This sub workflow is called from a cancelled workflow before the cancel transition is executed. It is advised not to change this workflow. |
| `checkrestart_subflowv1.xml` | Subflow to check if the parent event should be restarted. |
| `wfin24classimpv2.xml` | Workflow to import classification codes based on a datasource. |

| File | Description |
|---|---|
| `wfin26BackEndIntegrationV1_Sample1.xml` | Sample workflow for setting up integration with a backend system.<br>For more details, see *TIBCO MDM System Administration*. |
| `wfin26BackEndIntegrationV1_Sample2.xml` | Sample workflow for setting up integration with a backend system.<br>For more details, see *TIBCO MDM System Administration*. |
| `wfin26catactionv2.xml` | Publication request notification (RFCIN) workflow for GDSN. |
| `wfin26catmassupdate2v1.xml` | Mass update workflow. This is a sub flow called by `wfin26catmassupdatev1`. |
| `wfin26catmassupdatev1.xml` | Mass update workflow. |
| `wfin26catmultipartysynchv2.xml` | Workflow to kick off synchronization when more than one partner or backend system is selected for synchronization. This workflow iterates over each selected partner and initiates synchronization for each partner. |
| `wfin26dqcatsourceimportv1.xml` | Data quality workflow to process import records sequentially. |
| `wfin26catsourceimportv2.xml` | Workflow for import or load and import events. |
| `wfin26catsourcev4.xml` | Workflow for data source loads. |
| `wfin26catsourcev5.xml` | Workflow for data source loads and import of records. |
| `wfin26catsourcev6.xml` | Workflow for data source upload and import of records. |
| `wfin26catsourcev7.xml` | Workflow for data source upload and import of records. A new AddressCleansing activity is added in this workflow. |
| `wfin26catsynchdbdumpv2.xml` | Workflow to extract the master catalog data into a CSV file. The workflow does not create any history and does not support incremental extraction. |
| `wfin26catsynchdbdumpv3.xml` | Workflow to extract and process repository data. |

| File | Description |
|---|---|
| `wfin26catsynchdbdumpv4.xml` | Workflow to extract all relationships and related records. Used by the Export Records feature. You can customize the workflow for exporting the selected records using the EvaluateSubset activity. |
| `wfin26catsynchdbdumpv4_incremental.xml` | Workflow to export data in an incremental mode, that is, only records that are added or updated after the last export are exported. Used by the Incremental Export Records feature. |
| `wfin26catsynchv6.xml` | Data Synchronization workflow. |
| `wfin26catsynchv7.xml` | Data Synchronization workflow. |
| `wfin26dataservicev2.xml` | Workflow to import or export meta data into XML. |
| `wfin26dqcatsourcev1.xml` | Import workflow for data quality. |
| `wfin26dqproductaddapprovalv1.xml` | Data quality workflow for new record introduction through the UI. |
| `wfin26dqcatAddRecordApprovalv1.xml` | Data quality workflow for new record introduction with the match and merge operation. |
| `wfin26dqscheduleddedupv1.xml` | Scheduler duplicated detection workflow. |
| `wfin26prodpubreqapprovalv2.xml` | Synchronization request notification workflow. |
| `wfin26productaddapprovalv3.xml` | Workflow to implement record introduction, conflict resolution, and lights-out synchronization of a new record. |
| `wfin26productaddinternaleditv1.xml` | Invoked as a subflow process from `wfin26dqproductaddapprovalv1.xml`. |
| `wfin26productcorrectv1.xml` | Workflow to implement record correction. This workflow is applicable only for UI operation correction. |
| `wfin26producteditapprovalv3.xml` | Workflow to implement record modification, conflict resolution, and lights-out synchronization of changes. |
| `wfin26productfactsheetpdfv2.xml` | Sample workflow to demonstrate how to generate a PDF using translate activity. The workflow generate a PDF file for record data. |
| `wfin26productmanualmergev1.xml` | Workflow to execute manual merge process. |

| File | Description |
|---|---|
| `wfin26productmergeapprovalv1.xml` | Invoked as a subflow process from `wfin26dqcatsourcev1.xml`. |
| `wfin26purgev2.xml`<br>`wfin26purgev3.xml` | Workflow to execute data purge. |
| • `wfin26hierarchycreateapprovalv1.xml`<br>• `wfin26hierarchyeditapprovalv1.xml`<br>• `wfin26hierarchydeleteapprovalv1.xml` | Workflows to process approval of the hierarchy operations. The following activities are added in these workflows:<br>• GetHierarchy<br>• CheckHierarchyState<br>– LockHierarchy<br>– CheckLockState<br>• UpdateHierarchyState<br>The CreateWorkItem activity is updated for hierarchy approval, that is, for create, edit, delete, and link operations.<br>For more information, see *TIBCO MDM Workflow Reference*. |
| `wfin26hierarchylinkapprovalv1.xml` | Workflow to process approval of hierarchy link operation. The UpdateHierarchyLink activity is added in this workflow. |

## GDSN Specific Workflows

The workflows specific to GDSN are located in the `$MQ_HOME`/datapool/common/standard/workflow folder.

*GDSN Specific Workflows*

| File | Description |
|---|---|
| `wfin26RFCINv2.xml` | Workflow to receive and process publication request (RFCIN) for suppliers. |
| `wfin26catsynchv7_gdsn.xml` | Workflow to initiate synchronization with datapool or custom channel. |
| `wfin26cim2cimsyncV1.xml` | CIM to CIM synchronization workflow. |
| `wfin26proddatanotifbasicv2.xml` | Incoming record notification workflow for, primarily, data add/change messages. Recommended for GDSN retailers and similar usage where received data needs to be processed and saved. |

| File | Description |
|---|---|
| `wfin26prodnotifbasicv3.xml` | Incoming record notification workflow, primarily, for data publication messages. It is a variation of wfin26proddatanotifbasicv2. |
| `wfin26prodnotifprocessv2.xml` | Workflow to process incoming record message and save data. |
| `wfin26prodnotifretailerv4.xml` | Workflow to process incoming record message and save data. It is a variation of wfin26prodnotifprocessv2 targeted towards datapools and integration hubs. |
| `wfin26prodnotifv2.xml` | Workflow to receive a notification of a response for a previous synchronization request sent. Notification is issued by TIBCO MDM when a response is received and processed. |
| `wfin26catactionRFCINv2.xml` | Workflow to implement initiation of publication request (RFCIN) for retailers. |
| `wfout1sync62cic26v1.xml` | Workflow to translate incoming 1Sync CIC message to the mlXML format. |
| `wfoutagentrics50cin26v2.xml` | Workflow to translate incoming Agentrics CIN message to the mlXML format. |
| `wfoutveloselcin26v2.xml` | Workflow to translate incoming CIM2CIM message to the mlXML format. |

## Forms

The form files are located in the `$MQ_HOME/common/standard/forms` folder.

*Forms*

| File | Description |
|---|---|
| `fm26ca.xml` | Used to define data mappings for most of the e-mails. |
| `fm26catpubtemplatev1.xml` | Used to define data mappings to create a spawn workflow request for synchronization workflows. |
| `fm26importtemplatev1.xml` | Used to define data mappings to create a import approval workflow request for a record. Such workflows are initiated when import is done. |

## Catalogs

The catalogs are located in the `$MQ_HOME/common/standard/catalog/master` folder.

*Catalogs*

| File | Description |
|---|---|
| `CatalogProductValidator` | Sample custom record validator class. Out-of-the-box validations supported in the StandardCatalogProductValidator can be overridden to provide custom validations. |

## Miscellaneous Files

The miscellaneous files are located in the `$MQ_HOME/common/standard/misc` folder.

*Miscellaneous Files*

| File | Description |
|---|---|
| `subscriptionpack.zip` | This subscription pack contains catalog metadata and a SQL script required to handle RFCIN messages.<br>• A Jar file which contains:<br>– Master catalog and output map for catalog format EAN.UCC.Subscription. Master catalog will be associated with GPC predefined classification.<br>– The catalog which uses the EAN.UCC.Subscription and master catalog for RFCIN generation.<br>– Data source for import of subscription and corresponding input map.<br>– Input map for master catalog.<br>• Rulebase validation file for master catalog.<br>• A SQL script to create associations with the datapools and a sequence, MQ_SEQUENCE_SUBSCRIPTION, used in rulebase validation to assign unique value to PRODUCTID. |

## Sample Files

The sample files are located in the `$MQ_HOME/common/standard/samples` folder.

*Sample Files*

| Folder/File | Description |
|---|---|
| `/authentication/LdapHelper.JAVA`<br>`/authentication/rolecreation_sample.SQL` | Sample for LDAP authentication. |

| Folder/File | Description |
|---|---|
| `/BackEndIntegration_Using_BW/Sample1.zip`<br>`/BackEndIntegration_Using_BW/Sample2.zip` | The following samples demonstrate how to integrate TIBCO MDM with TIBCO BusinessWorks using JMS.<br>• **Sample1.zip - Inbound Integration**: In this sample, BusinessWorks sends a JMS message to add a record in TIBCO MDM and receives the response message on the outbound queue which can be consumed by BusinessWorks to verify if the record was added successfully.<br>For more details, refer to Chapter 3 "Integration with TIBCO Business Works - Sample 1" of the *TIBCO MDM System Administrator's Guide*.<br>• **Sample2.zip - Outbound Integration**: When a record add event occurs in TIBCO MDM, the workflow sends a JMS message as an mlXML document wrapped in ebXML payload to BusinessWorks and the workflow is suspended till it receives a notification from BusinessWorks. Once the notification is received, the workflow is successfully completed.<br>For more details, refer to Chapter 4 "Integration with TIBCO Business Works - Sample 2" of the *TIBCO MDM System Administrator's Guide*. |
| `/BC-BW-Configuration/SampleBC-Configuration.csx`<br>`/BC-BW-Configuration/SampleBWConfiguration.zip` | This sample demonstrates how to integrate TIBCO MDM with datapools like WWRE and 1Sync using TIBCO BusinessConnect and TIBCO BusinessWorks.<br>For more details, refer to Chapter 7 "Configure TIBCO BusinessConnect and TIBCO BusinessWorks" of the *TIBCO MDM Installation and Configuration Guide*. |
| `/CIM Repackaging and code Reorg/8.0 Repackaging snapshot.xls`<br>`/CIM Repackaging and code Reorg/Repackaging impact on workflows and rules` | Sample documents to show repackaging impact. |

| Folder/File | Description |
|---|---|
| `/configuration/ insertConfigDefinition_sample.sql` | This sample demonstrates how to add a rulebase file entry to the ConfigurationDefinition table. The rulebase added here is executed on the record before it is published. The rulebase file must be present in the `$MQ_COMMON_DIR/mydir/rulebase` folder.<br>Edit the script to change the data as per your business requirement.<br>Connect to the database and run the following script:<br>sqlplus user/passwd@connect_string @insertConfigDefinition_sample.sql |
| `/CustomResponseHandler/ CustomResponseHandler.java` | This sample demonstrates how to use more than one registration key to queue the event accordingly.<br>Refer to */CustomResponseHandler/Configuring Custom Handler.txt* for more information. |
| `/EmailWorkitemHtmls/*.html` | Sample workitem emails that can be customized. |
| `/Java Transitions/ CustomConditionEval.java`<br>`/Java Transitions/ wfin26productadddeletev1.xml` | This workflow sample XML demonstrates how to use Java-based transitions.<br>Compile the CustomConditionEval.java file and place the generated class in the veloselpatches folder.<br>In this class, a rule is defined which specifies the condition in the Java format. The CDATA section contains the custom condition com.tibco.eval.CustomConditionEval.translateJmsToMLToDeleteRecord(Mcname) to be evaluated.<br>In this sample, if the master catalog name is "asset", the "DeleteRecord" activity is executed. If the master catalog name is not "asset", the "SaveRecord" activity is executed. Here is how it is implemented:<br>• The method translateJmsToMLToDeleteRecord evaluates to true only if the MasterCatalogName is equal to "asset", so when the result returned is true, "DeleteRecord" Activity is executed. Based on the result returned, further transition path is decided.<br>• The method translateJmsToMLToSaveRecord evaluates to true only if the MasterCatalogName is not equal to "asset". So, if the result returned is true, the "SaveRecord" Activity is executed. |

<table>
<tr><th>Folder/File</th><th>Description</th></tr>
<tr><td>

```
/JSXAPPS/CIMTestGIProject/*
```

</td><td>

This project allows you to build custom screens for various operations such as the following using web services:

1. Record Add

2. Record View

3. Record Modify

4. Inbox (work item list)

5. Work Item detail

Refer to the */JSXAPPS/readme.txt* for more information.

</td></tr>
<tr><td>

```
/rulebase/connectsample.xml
/rulebase/disconnectsample.xml
/rulebase/queryanothercatalog
```

</td><td>

This sample demonstrates how to create a "Shipping Address" relationship between two catalogs: Customer and Address.

The rulebase declares two variables:

```
ADDRESSCATALOG link type="catalog"
ADDRESSRECORDS link type="record"
```

The constraint "AddressSoftlink" returns ADDRESS records having same CUSTOMERID as that of the record being processed.

The constraint "AddressConnect" connects ADDRESS records with the record being processed using the relationship "Shipping Address". The following syntax is used:

```
<constraint>
    <name>AddressConnect</name>
    <description>
        Connect Address records using
'Shipping Address' relationship
</description>
    <action>
        <connect>
            <literal>Shipping Address</
literal>
<!--relationship name -->
            <var>ADDRESSRECORDS</
var>
            <literal>10</literal>
<!--quantity/optional/default 0-->
        </connect>
    </action>
</constraint>
```

</td></tr>
<tr><td>

```
rulebase/RulebaseCustomFunction.java
```

</td><td>

This sample code defines the interface for third party product validators.

The code generates a product ID and assigns it to the productid attribute if it does not exist. It then proceeds to call Standard validations.

</td></tr>
</table>

| Folder/File | Description |
|---|---|
| `rulebase/nested/` | The files in this folder demonstrate how to embed one rulebase into another.<br>You can embed a rule in the following two rulebase format files:<br>`CatalogValidation.xml`: In the `catalogvalidation.xml` file the following rulebase exists:<br>`<constraint>`<br>`<name>Include embed1</name>`<br>`<description>RECORD_TYPE dropdown</`<br>`description>`<br>`<action>`<br>`<include>`<br>`<literal>/standard/rulebase/embed1.xml</`<br>`literal>`<br>`    </include>`<br>`</action>`<br>`</constraint>`<br>`NewRecord.xml`: The new record is embedded in `newrecord.xml`.<br>`<constraint>`<br>`<name>Include newrecord_embed</name>`<br>`<description>default propagation</`<br>`description>`<br>`<action>`<br>`<include>`<br>`<literal>/standard/rulebase/`<br>`newrecord_embed.xml</literal>`<br>`<literal>embed</literal>`<br>`</include>`<br>`</action>`<br>`</constraint>` |
| `/scripts/Create_Purge_Package.sql` | This script creates a SQL package that is used to purge all redundant data on TIBCO MDM 7.x installation. The SQL package contains the procedure PurgeHistoricalData, which actually purges the data.<br>For more information, refer to the usage guidelines documented in the SQL script. |
| `scripts/denyAccessForAFunction.sql` | This script shows how access to a new function may be denied for selected roles .<br>For more information, refer to the usage guidelines documented in the SQL script. |
| `/workflow/Merge_Match_Record_Sample/`<br>`MatcherWorkItem.html`<br>`MatcherWorkItemServlet.java`<br>`wfin26productaddapprovalv3.xml`<br>`wfin26productaddinternaleditv1.xml`<br>`rulebase/skipAttrlist.xml` | The folder contains files that demonstrate how to implement data quality using the match and merge records functionality.<br>The `MatcherWorkItem.html` and MatcherWorkItemServlet.java are used for the data quality work item page. You can customize the work item page. For more details, refer the *TIBCO MDM Customization Guide.* |

| Folder/File | Description |
|---|---|
| workflow/<br>FilesAndDirectoriesCleaner.java | This sample code demonstrates how to customize purging. The workflow wfin26purgev3.xml has a PurgeFilesThroughShellScript activity which calls com.tibco.mdm.workflow.engine.activities.FilesAndDirectoriesCleaner.purgeWorkDirectoryContent(purgefilepath,waitForPurgeProcessFlag).<br>This method purges all physical files for the redundant GeneralDocument entries using an OS-specific script.<br>The script is generated by the Purge activity (DocumentHandler.generateDocumentList()).<br>• @param purgeFilePath — Relative path of the file having list of the files to be purged.<br>• @param waitForPurgeProcess — A flag to identify whether the child process which executes the script should wait for parent process. The default value is false.<br>The following is sample activity that calls the method FilesAndDirectoriesCleaner.purgeWorkDirectoryContent.<br>&lt;Activity Name="PurgeFilesThroughShellScript"&gt;<br>&lt;Action&gt;InterpretCommand&lt;/Action&gt;<br>&lt;Description lang="en"&gt;Delete the files associated with purged data&lt;/Description&gt;<br>&lt;Parameter direction="in" eval="variable" type="string" name="purgefilepath"&gt;relativepath&lt;/Parameter&gt;<br>&lt;!-- &lt;Parameter direction="in" eval="constant" type="boolean" name="waitForPurgeProcess"&gt;true&lt;/Parameter&gt; --&gt;<br>&lt;Script format="bsh"&gt;&lt;![CDATA[<br>java.lang.Boolean waitForPurgeProcessFlag = null;<br>if(waitForPurgeProcess != void)<br>waitForPurgeProcessFlag = waitForPurgeProcess;<br>com.tibco.mdm.workflow.engine.activities.FilesAndDirectoriesCleaner.purgeWorkDirectoryContent(purgefilepath,waitForPurgeProcessFlag);<br>&lt;/Script&gt;<br>&lt;/Activity&gt; |

| Folder/File | Description |
|---|---|
| `workflow/ MqActivityInstPostProcess1.java` | This sample demonstrates how to write a custom workflow activity.<br>Every custom activity must be derived from MqActivityInstImpl and must provide the custom implementation in the "execActivity" method. The activity takes `masterCatalogID` as input parameter (which is extracted from MqWorkflowState) and the PostProcess method is called which in turn calls the stored procedure post_processor.<br>The method getRequiredParameters() returns the array of parameters required by this activity.<br>The workflow engine validates the input parameter list before the activity is called.<br>The method getOptionalParameters() returns the optional parameters. The list of optional parameters is used by workflow to ensure that no parameter values are carried over from a previous activity.<br>For example, if ReferenceStepID is not defined in the current activity, but was defined for a previous activity, this parameter has to be reset to 'null'. Otherwise, the value of ReferenceStepID used in the current activity will be the value carried over from the previous activity. |
| `workflow/ wfin26productaddapprovalv3.xml` | This is sample workflow that demonstrates "product add approval". It routes the Product Add to editors and approvers.<br>Following are the main steps involved in this workflow:<br>InternalEdit — Uses 'New Product Introduction Edit' rules to identify editors.<br>InternalApproval — Uses 'Product Edit Approval' rules to identify approvers.<br>TargettedRejection — Uses 'Data Custodian' rules to identify data custodians. |

| Folder/File | Description |
|---|---|
| `workflow/wfin26Test1.xml` | This sample workflow demonstrates how to spawn a subflow. The following is the sample fragment for spawning the subflow.<br>`    <Activity Name="TestSubflow">`<br>`        <Action>NoOperation</Action>`<br>`        <Description>Test Subflow</`<br>`Description>`<br>`        <Execution>ASYNCHR</Execution>`<br>`<!-- Assign some more values -->`<br>`        <Parameter direction="in"`<br>`type="string" eval="constant"`<br>`name="eventState">SPAWNWORKFLOW</`<br>`Parameter>`<br>`        <Parameter direction="in"`<br>`type="document" eval="variable"`<br>`name="InDocument">inDoc</Parameter>`<br>`     </Activity>`<br>Activity implementation for Noop activity: The Noop activity expects no IN parameters and returns the same state. It is supplied primarily for routing when an empty workflow or an activity that does nothing is required. For example, a Noop activity can be useful in split/join transitions. |