



TIBCO BusinessEvents® Enterprise Edition

Decision Manager User Guide

*Version 6.3.0
July 2023*



Contents

Contents	2
Before You Begin	6
Rule Management Server Prerequisite	6
Third-Party Software Documentation References	7
TIBCO BusinessEvents Decision Manager Overview	9
Decision Manager Without RMS	9
Technical User Tasks	10
Decision Manager and RMS User Workflow	11
Decision Manager User Interface	12
Configuring Decision Manager and RMS	15
Enabling Remote Connection to RMS from Decision Manager	15
Defining the RMS Server Host Location	15
Setting RMS Server Host Location in Decision Manager	16
Configuring One-Way SSL Authentication Between RMS Server LDAP Server and Decision Manager Client	17
Configuring SSL-Based HTTP Communication Between Decision Manager And Rule Management Server	19
Configuring the Decision Manager Component Properties	20
Decision Manager Component Property Reference	20
Setting Application Preferences	21
Decision Table Related Preferences	22
Working with RMS Projects and Artifacts in Decision Manager	27
Starting and Logging in to RMS	27
Checking Out a Project	28
Updating (Synchronizing) a Project	29

Committing Artifacts	30
Committing Project Artifacts to RMS at the Command Line	31
Viewing the History of RMS Artifacts	33
The History View Reference	34
Comparing and Merging Artifacts	35
Reverting to a Master Copy	37
Rule Building with Decision Tables	39
Working with Decision Table	40
Adding a Decision Table	40
Setting Decision Table Properties	41
Viewing Decision Trees	41
Deleting Decision Tables	42
Setting the Decision Table Effective Period	43
Comparing and Merging Decision Tables	44
Compare Editor Buttons	45
Columns and Rows (Rules) in a Decision Table	46
Working With Decision Table Rules (Rows) and Columns	48
Adding Decision Table Condition and Action Columns	49
Defining Custom Columns Using Substitution	50
Arrays in Columns	51
Moving Resizing and Sorting Columns	53
Setting Default Value in Decision Table Column	54
Deleting Rows and Columns	54
Duplicating Rules (Rows)	55
Merging Rows	55
Setting Rule (Row) Priorities	56
Filtering Rows	56
Domain Models in Table Cells	57
Catalog Functions in Table Cells	57
Setting the Date-Time Values Using the Calendar Dialog	58
Supported Operators	59

Exporting and Importing Decision Tables	61
Exporting a Decision Table to a Microsoft Excel File	61
Exporting Decision Tables to the Microsoft Excel Files at the Command Line	62
Importing the Microsoft Excel File to a Decision Table	64
Importing the Microsoft Excel File to a Decision Table at the Command Line	65
Converting a CSV File to a Microsoft Excel File with the Studio Tools Utility	66
Microsoft Excel File Format for Decision Tables	67
Decision Tables Validation	70
The Analyze Table Tool	71
The Show Coverage Tool	72
The Generate Test Data Tool	73
Auto-Fixing Issues	73
Setting Table Analyzer Preferences	73
Validating a Decision Table Using Test Data	74
Validating a Decision Table at the Command Line	75
Generating and Deploying Decision Table	77
Configuring for Deployment and Unloading of Decision Table Classes	77
Configuring for Loading and Deployment of Decision Table Classes	78
Configuring for Hot Deployment and Unloading of Decision Table Classes	80
Configuring RMI Properties for Remote Invocation	83
Generating Deployable Files (EAR and Class Files)	84
Generating the Project EAR or All Project Class Files	85
Generating One Decision Table's Class File at the Command Line	86
Generating All Project Class Files at the Command Line	88
Deploying and Unloading Decision Table Class Files	91
Hot Deploying External Classes Using JMX	92
Advisory Events for Deployment Success and Failure	92
Generated Files Location	93
TIBCO Documentation and Support Services	95

Legal and Third-Party Notices	98
--	-----------

Before You Begin

To maintain uniformity, the following terms have been used in the TIBCO BusinessEvents Studio UI and the product documentation:

- TIBCO ActiveSpaces software version 2.x is referred to as *Legacy ActiveSpaces*.
- TIBCO ActiveSpaces software version 4.6.1 and later are referred to as *ActiveSpaces*.

For details about the supported versions, see the *Readme.txt* file available at the [TIBCO BusinessEvents® Enterprise Edition Product Documentation](#) page.

Rule Management Server Prerequisite

In addition to Legacy ActiveSpaces as cluster and cache provider, you can also configure TIBCO BusinessEvents Rule Management Server (RMS) with the following combinations:

Cluster	Cache	Store
Apache Ignite	Apache Ignite	None/Shared Nothing/RDBMS/Store Providers (TIBCO ActiveSpaces and Cassandra)
TIBCO FTL	Apache Ignite	None/Shared Nothing/RDBMS/Store Providers (TIBCO ActiveSpaces and Cassandra)
TIBCO FTL	No cache	TIBCO ActiveSpaces

By default, Apache Ignite is used as the cluster and cache provider.

For more information about configuring these for your RMS project, see *TIBCO BusinessEvents Configuration Guide*.

Third-Party Software Documentation References

For complete details about the third-party software used in the project, see its documentation.



Note: When you obtain third-party software or services, it is your responsibility to ensure you understand the license terms associated with such third-party software or services and comply with such terms.

Third-Party Software Documentation

Software	Used as	Documentation Reference URL
TIBCO ActiveSpaces 4.6.1 and above	Store provider	TIBCO ActiveSpaces documentation
TIBCO ActiveSpaces 2.x	Cluster and Cache provider	TIBCO ActiveSpaces documentation
Apache Kafka	Channel	Apache Kafka documentation
Confluent Schema Registry	Schema Registry	Confluent documentation
TIBCO Messaging - Schema Repository for Apache Kafka	Schema Registry	TIBCO Messaging - Schema Repository for Apache Kafka documentation
Apache Pulsar	Channel	Apache Pulsar documentation
Apache Cassandra	Store provider	Apache Cassandra documentation
GridGain	Data Center Replication	GridGain documentation
TIBCO FTL	Cluster provider	TIBCO FTL documentation

Software	Used as	Documentation Reference URL
Apache Ignite	Cluster and Cache provider	Apache Ignite documentation
InfluxDB	Metrics store provider	InfluxDB documentation
Grafana	Application metrics visualization	Grafana documentation

TIBCO BusinessEvents Decision Manager Overview

TIBCO BusinessEvents Decision Manager uses the Rules Management Server (RMS) provided with TIBCO BusinessEvents.

Decision Manager Component

The Decision Manager component adapts TIBCO BusinessEvents Studio for the purpose of enabling business users to create and populate decision tables, that are then deployed in a TIBCO BusinessEvents project. The Decision Manager provides graphical environment for business users which supports and constrains their actions. TIBCO BusinessEvents Decision Manager supports access control to define what users with different user roles can do in the system.

See [Decision Manager User Interface](#).

Rules Management Server (RMS) Component

Rules Management Server (RMS) is a server-based component that manages the lifecycle of Decision Manager artifacts and Decision Manager user access.

Using a menu of options in TIBCO BusinessEvents Studio, RMS allows business users to check out projects and check them in, subject to an approval process that RMS also provides. The workflow process can be customized. For a more detailed overview see [Decision Manager and RMS User Workflow](#).

Before RMS can be used, some configuration is required, see *TIBCO BusinessEvents WebStudio User Guide* of the TIBCO BusinessEvents documentation set for details.

Decision Manager Without RMS

Use of RMS is optional, unless TIBCO BusinessEvents WebStudio is used. TIBCO BusinessEvents WebStudio requires RMS.

If file system access to project resources is provided, Decision Manager users can use project import options instead of using RMS menu options. Decision tables can also be created and maintained in TIBCO BusinessEvents Studio.

If you use Decision Manager without RMS, the following features are unavailable:

- Log in and log out
- Check out, Update, Commit, and Revert
- Show work list, including the related Approve, Reject, Deploy options.
- Show history
- Generate deployable
- You can deploy and hot-deploy decision tables as part of a TIBCO BusinessEvents project EAR, but not as class files.

Technical User Tasks

As a technical user you configure RMS and the Decision Manager components, and set up project resources required for business users to create decision tables.

You also deploy decision tables when they are ready for use, either in an enterprise archive (EAR) file, or as class files. Class files can be deployed for use at startup, or hot deployed. They can also be unloaded from a running engine. You also define access control settings and can act as an approver to ensure that decision tables created by business users are appropriate for use.



Note: Decision Manager is supported only on Linux and Windows. The server component (RMS), however, is supported on the same platforms as TIBCO BusinessEvents.

Virtual Rule Functions and Decision Tables

At design time, technical users add *virtual rule functions* (VRFs) to a TIBCO BusinessEvents project. A VRF has no body, similar to a Java interface. Its implementation is provided using decision tables authored in Decision Manager. VRFs are used in the TIBCO BusinessEvents project like any other rule function; they can be called from rules or other rule functions. Here is a simple example:

```

/**
 * @description
 */
virtual void rulefunction Virtual_RF.Applicant_VirtualRuleFunction {
    attribute {
        validity = ACTION;
    }
    scope {
        Concepts.Applicant      applicant;
        Events.ApplicationReceived applicationreceived;
    }
    body {
    }
}

```

A decision table implemented for the specified VRF is shown in [Decision Manager User Interface](#).

In Decision Manager, business users add decision table resources to VRFs. The decision table provides the body to the VRF, also known as the VRF implementation.

One VRF can have more than one decision table. If a VRF has more than one decision table, functions in TIBCO BusinessEvents determine how the tables are used.

Decision Manager and RMS User Workflow

The rules authored in TIBCO BusinessEvents Decision Manager follows a approval workflow, before they can be deployed.

See *TIBCO BusinessEvents WebStudio User Guide* of the TIBCO BusinessEvents documentation set for complete approval workflow in RMS.

Step 1: Setup Project in TIBCO BusinessEvents

A TIBCO BusinessEvents user creates a TIBCO BusinessEvents project, adding the ontology, and building decision tables that make use of virtual rule functions (VRFs). At design time, VRFs have a signature but no body.

The TIBCO BusinessEvents project for RMS is stored at the location defined by a property in the RMS server configuration. The RMS project requires an access control file.

The RMS server must be running so that the log in and workflow actions are available in the Decision Manager component.

Step 2: Create Decision Table in TIBCO BusinessEvents Decision Manager

A business user starts the Decision Manager component, logs on to RMS using the RMS menu option, and checks out the project. The business user creates one or more decision tables and saves the modified project locally, then commits them for approval.

Step 3: Approval

An RMS user working in TIBCO BusinessEvents WebStudio receives the request and reviews the checked-in artifacts and then approves or rejects them. The approved artifacts are available for subsequent checkouts or updates.

Step 4: Generating Deployable Files

An RMS user generates deployable files for resources that are ready for deployment. You can generate EAR files or class files.

Step 5: Deployment

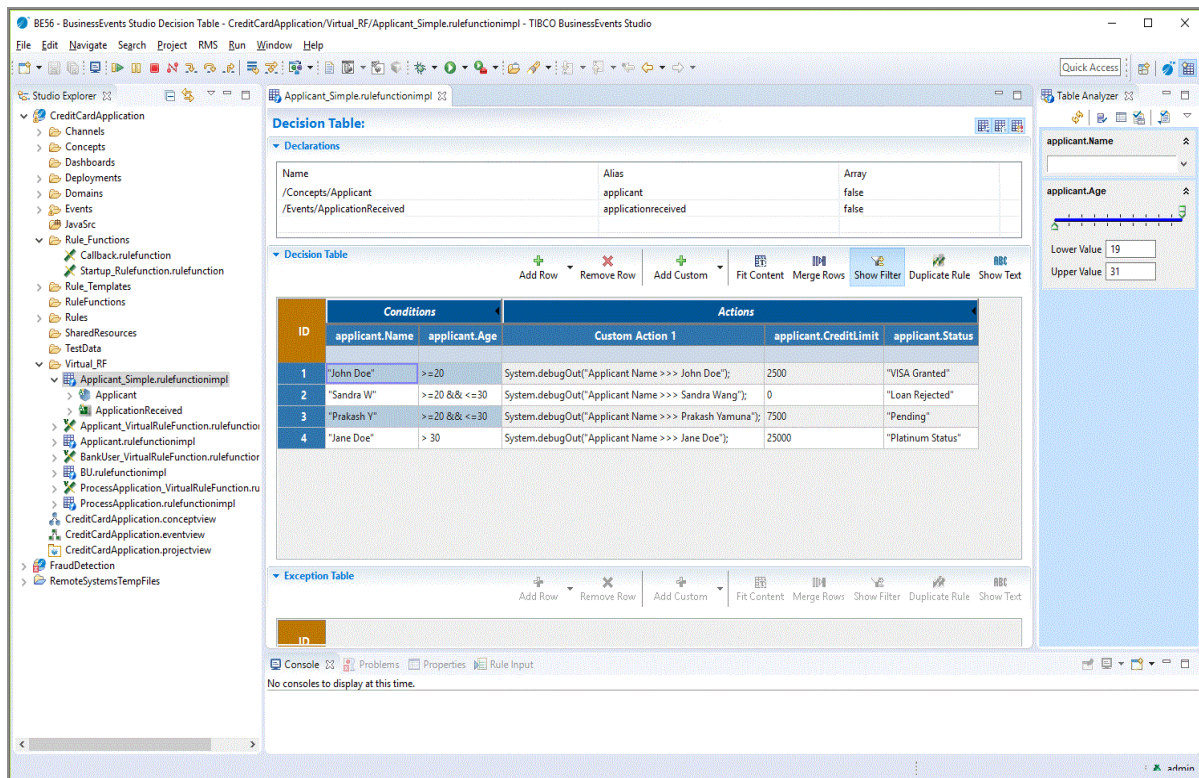
EAR files are deployed in the usual way, as explained in *TIBCO BusinessEvents Administration* of the TIBCO BusinessEvents documentation set. Class files for decision tables are deployed by placing them in a configured location recognized by the TIBCO BusinessEvents engines at startup. They can also be hot-deployed.

Decision Manager User Interface

The TIBCO BusinessEvents Decision Manager user interface consists of TIBCO BusinessEvents Studio Explorer and the decision manager editor.

The following figure displays the decision table that is implementation of the VRF shown in the following figure.

Decision Manager User Interface



Tip: The following are few navigation tips for the Decision Manager user interface:

- Select **Window>Show View** to reopen views you closed.
- Select **Window>Navigation** to navigate through various editors.

TIBCO BusinessEvents Studio Explorer

In TIBCO BusinessEvents Decision Manager, TIBCO BusinessEvents Studio Explorer (on the top-left) displays the TIBCO BusinessEvents ontology. The ontology is used by the business user to work with a decision table. Ontology is created in the TIBCO BusinessEvents project, where the decision tables you add are used.

Decision Table Editor

The editor is the work area for defining the decision tables. The editor consists of three different sections that can be hidden or viewed:

Declarations

Displays the VRF arguments (read-only).

Decision Table

The table used to define business rules by dragging and dropping properties of the VRF arguments (or using other logic).

Exception Table

Another instance of a decision table where you can (optionally) enter conditions and actions for non-business logic.

Overview View

The Overview view (on the bottom-left) displays the thumbnail of the decision table. It is used to navigate larger tables.

Catalog Functions

Three catalogs of functions are available: Standard Functions, Ontology Functions, and Custom Functions. Expand each set to view the functions. You can drag and drop functions onto the desired decision table cell to help you define a condition or action.

When you float the cursor over a function in the registry, Decision Manager displays a tooltip showing the description and syntax.

Custom Functions

Technical users can create custom functions (with their own tooltips). For more details, see *TIBCO BusinessEvents Developer's Guide* of the TIBCO BusinessEvents documentation set.

Configuring Decision Manager and RMS

Artifacts need to be added to RMS, so that they can be used by TIBCO BusinessEvents Decision Manager. Advanced configuration might also be required to enable certain features, such as, SSL authentication.

Enabling Remote Connection to RMS from Decision Manager

RMS works out-of-the-box on a local machine. Configuration is required to enable access by remote Decision Manager clients.

You may need to make additional changes to the server configuration. For example if you change the server location or the location of the RMS project repository, you must update RMS server properties accordingly.

To configure RMS server properties, you edit the `RMS.cdd` file. See *TIBCO BusinessEvents WebStudio User Guide* of the TIBCO BusinessEvents documentation set for information on configuring RMS Server properties.

Defining the RMS Server Host Location

Define the RMS server hostname and port to ensure that Decision Manager users can connect to RMS from remote machines.

Setting the `tibco.clientVar.RMS/hostname` property to `localhost` enables the product to be used on a single machine. For additional Decision Manager configuration options, see [Configuring the Decision Manager Component Properties](#).

i Note: TIBCO BusinessEvents Decision Manager clients must have direct network connectivity to the RMS server. Connections from clients may not work if the RMS server is behind a firewall.

Procedure

1. Import the BRMS project into your workspace and open the `RMS.cdd` file for editing. See *TIBCO BusinessEvents WebStudio User Guide* of the TIBCO BusinessEvents documentation set for details.
2. In the **CDD editor Processing Units** tab, click **Inference**. In the Properties panel on the right, expand the RMS-GVs property group.
3. Specify the RMS hostname and port using the following properties:

```
tibco.clientVar.RMS/hostname
tibco.clientVar.RMS/port
```

4. Save the `RMS.cdd` file
5. In the file system, copy the `RMS.cdd` file from your workspace to the BRMS project and to the RMS server startup location.
 - BRMS project location: `BE_HOME/rms/project/BRMS/Deployment/RMS.cdd`
 - RMS server location: `BE_HOME/rms/bin/RMS.cdd`
6. Restart the RMS server.

Setting RMS Server Host Location in Decision Manager

Specify the location of the host in the startup file for the Decision Manager component, so that Decision Manager can connect to RMS.

The `studio.tra` file properties contains the default connection values. They are displayed when users log in to RMS, using a menu option in the Decision Manager user interface. Users can also specify different values to connect to another server.

Procedure

1. Open the following file for editing:
`BE_HOME/studio/eclipse/configuration/studio.tra`

2. Set the following properties to use the same hostname and port as you defined in the section [Defining the RMS Server Host Location](#):

```
#RMS interaction properties  
rms.host=hostname  
rms.port=port
```

3. Save the file.

Configuring One-Way SSL Authentication Between RMS Server LDAP Server and Decision Manager Client

One-way SSL authentication enables the application operating as the SSL client to verify the identity of the application operating as the SSL server (RMS). The SSL-client application is not verified by the SSL-server application.

i Note: Authentication options shared by TIBCO BusinessEvents and its add-on products, such as configuring file-based and LDAP authentication, are documented in *TIBCO BusinessEvents Administration* guide of the TIBCO BusinessEvents documentation set.

i Note: After configuration, if you change from secure to non-secure mode or from non-secure to secure mode, you must change the **be.auth.ldap.port** value and restart the RMS server.

Procedure

1. Ensure that the keystores are in place in the following locations, as required for the secure authentication you are setting up:
 - The RMS server machine
 - The LDAP server machine
 - All TIBCO BusinessEvents Decision Manager (that is, client) machines

2. Import the BRMS project into your workspace and open the `RMS.cdd` file for editing. See *TIBCO BusinessEvents WebStudio User Guide* of the TIBCO BusinessEvents documentation set for details.
3. In the CDD editor Cluster tab, click **Properties**.
4. To configure one way SSL between the RMS server and the LDAP server, do the following:
 - Add the properties for One-Way SSL between LDAP server, RMS server, and Decision Manager clients:

```
be.auth.ldap.ssl  
javax.net.ssl.trustStore  
javax.net.ssl.trustStorePassword  
javax.net.ssl.trustStoreType
```

See *TIBCO BusinessEvents WebStudio Users Guide* of the TIBCO BusinessEvents documentation set for `RMS.cdd` properties reference.

- Change the value of **be.auth.ldap.port**. Edit the property to specify the secure port (which is different from the non-secure port).
5. Save the `RMS.cdd` file.
 6. In the file system, copy the `RMS.cdd` file from your workspace to the BRMS project and to the RMS server startup location:
 - BRMS project location:
`BE_HOME/rms/project/BRMS/Deployment/RMS.cdd`
 - RMS server location:
`BE_HOME/rms/bin/RMS.cdd`
 7. Restart the RMS server.

Configuring SSL-Based HTTP Communication Between Decision Manager And Rule Management Server

Using one-way SSL authentication, you can verify the identity of the Decision Manager client from the Rule Management Server (RMS).

Procedure

1. Open the `RMS.cdd` file for editing.
The file is located at `BE_HOME/rms/bin/RMS.cdd`.
2. In the Default PU, edit the `be.channel.deactivate` property and remove the `/Transports/Channels/AMS_CH_ArtifactSecureCommunicationHTTPChannel` entry from its values.
3. Save the `RMS.cdd` file.
4. Start the RMS server.
5. Open the `decisionmanager.ini` file for editing.
The file is located at `BE_HOME/studio/eclipse/studio.ini`.
6. Add the following Java property in the file:

```
-Djavax.net.ssl.trustStore=<BE_HOME>\rms\config\security\SSLKeyStore.ks
```

Replace `BE_HOME` with the actual path.

7. Start Decision Manager.
8. In Decision Manager, click **RMS > Login** and enter `https://localhost:8443/Transports/Channels/AMS_CH_ArtifactSecureCommunicationHTTPChannel/` in **Management Server URL**. Log in using the RMS **Username** and **Password**.

Configuring the Decision Manager Component Properties

Configure the `studio.tra` file to alter the default configuration of the TIBCO BusinessEvents Decision Manager component.

See [Enabling Remote Connection to RMS from Decision Manager](#) to enable the remote connection to RMS.

Note: Restart TIBCO BusinessEvents Studio after making any changes to the `studio.tra` file.

Procedure

1. Open `BE_HOME/studio/eclipse/configuration/studio.tra`
2. Change the properties as needed.
See [Decision Manager Component Property Reference](#) for information about each property.
3. Save the file.

Decision Manager Component Property Reference

Edit the TIBCO BusinessEvents Studio `studio.tra` file to set the RMS connection properties.

TIBCO BusinessEvents Decision Manager Configuration Properties

Property	Notes
<code>rms.host</code>	Specifies the host name of the machine where RMS is running. Use the same host name with which RMS is started. See Enabling Remote Connection to RMS from Decision Manager . Default is <code>localhost</code>.

Property	Notes
<code>rms.port</code>	Specifies the port number for the RMS server. (See <code>rms.host</code> property notes for more details.) Default is 5000.
<code>rms.context.root</code>	Specifies the URL on which RMS is listening to requests over HTTP transport. (In TIBCO BusinessEvents terms, this is the channel on which the TIBCO BusinessEvents RMS application listens.) Default is <code>/Transports/Channels/AMS_CH_ArtifactCommunicationHTTPChannel/</code>
<code>rms.request.timeout</code>	Timeout value in milliseconds for RMS interactions. Default is 0 (zero).
<code>rms.heartbeat.delay</code>	Delay (in seconds) with which the Decision Manager component pings the RMS server to check status. Default is 10.
<code>rms.heartbeat.stop.elapsedtime</code>	Time interval (in minutes) till which the Decision Manager component pings the RMS server to check status. The Decision Manager component stop pinging after this interval. Default is 10.

Setting Application Preferences

You can change the appearance of Decision Manager by setting your own preferences. You can also change other preferences that affect the appearance and behavior of TIBCO BusinessEvents Studio, the user interface for Decision Manager.

All preferences are shown in *TIBCO BusinessEvents Developer's Guide* of the TIBCO BusinessEvents documentation set.

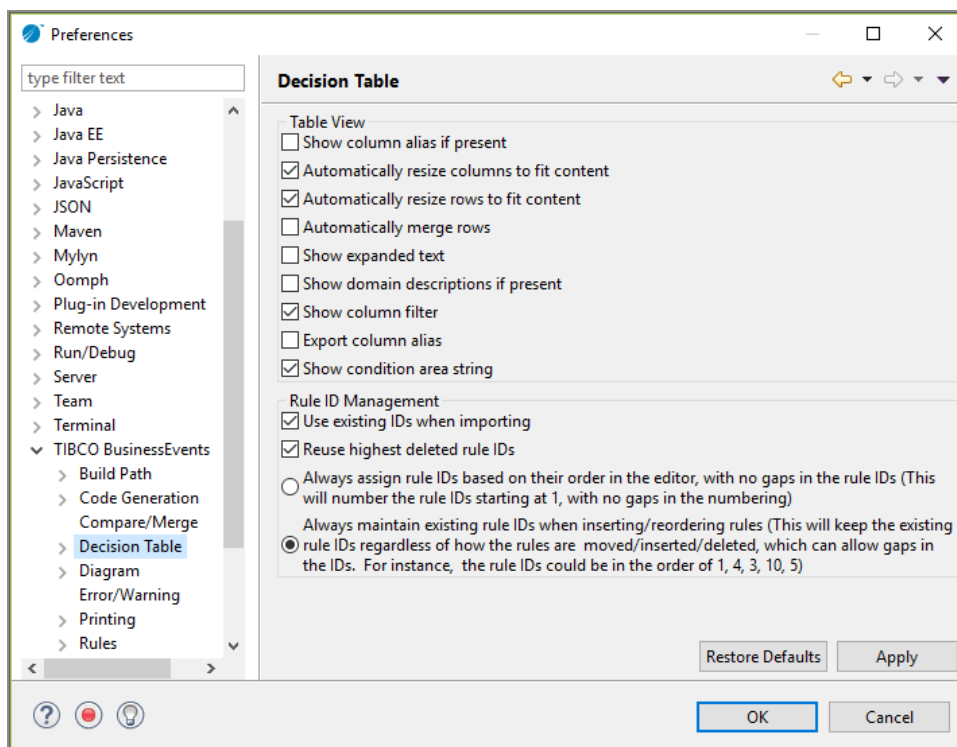
Procedure

1. From the Window menu, select **Preferences**.
2. Expand the options on the left and select each option to view a panel of options on the right.
Details on each set of options are provided in [Decision Table Related Preferences](#).
3. Change the options from the list as desired, and then click **Apply**.
4. Click **OK** to see Decision Manager with your preferred settings.

Decision Table Related Preferences

You can change the preferences related to Decision Manager from the preferences given in the TIBCO BusinessEvents Studio.

Decision Table View Preferences



The following Table View preferences set the initial state of the decision table. Text in parentheses shows the names of corresponding buttons in the Decision Table editor. The settings can be toggled in the UI, but their initial state is set by preferences you set.

- Show column alias if present
- Automatically resize columns to fit content (Fit Content button)
- Automatically resize rows to fit content
- Automatically merge rows (Merge Rows button)
- Show expanded text (Show Text button)
- Show domain descriptions if present.
- Show column filter
- Show titled border
- Use existing IDs when importing
- Export column alias
- Show condition area string

Decision Table Analyzer Preferences

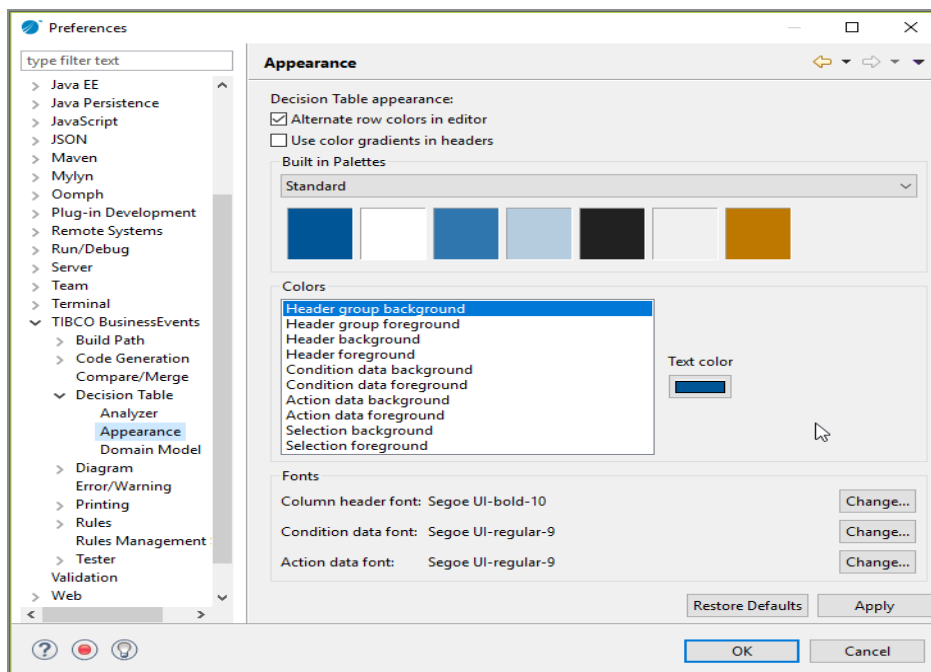
You can control the following behavior of the decision table analyzer, as well as its appearance:

Option	Description
Initially Show Table Analyzer View	Check box to show or hide the Table Analyzer view whenever a decision table is displayed. Also, to highlight rules in a decision table, which can be triggered on selecting the values in the analyzer view.
Highlight Partial Ranges	Check box to highlight rows whose condition falls completely or partially within the specified condition range.
Use Domain Model for Table Completeness	Select the check box to report any uncovered domain entries.
Analyze while opening table	Check box to analyze the decision table while opening it. • If there is any equal set of conditions but with different set of actions, the analyzer will report it.

Option	Description
	If multiple rules can be combined into one, the analyzer will report them.

Decision Table Appearance Preferences

These preferences let you set the color scheme for decision tables. For each item in the list, you can set color preferences. For condition data and action data you can specify a font.



Domain Model Preferences

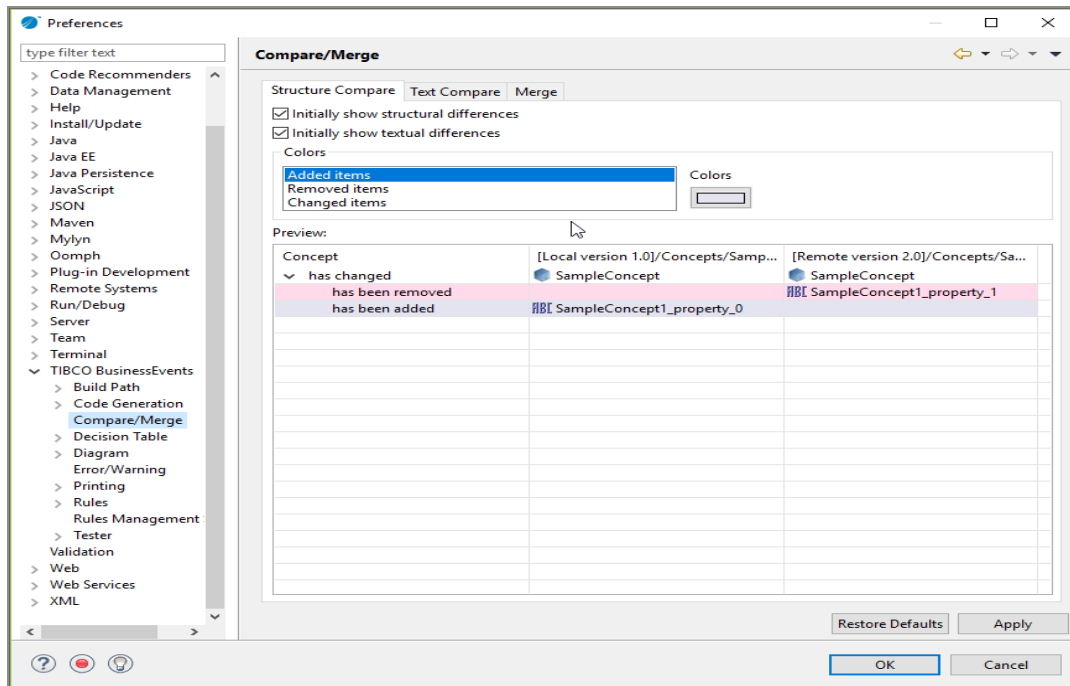
You can set the domain model preferences for the decision tables using this tab. The following options are available under the Domain Model preferences.

Option	Description
Allow custom values	If selected, custom values can be entered for the attributes associated to the domain. If the check box is cleared, you can only select from the domain values drop-down list.

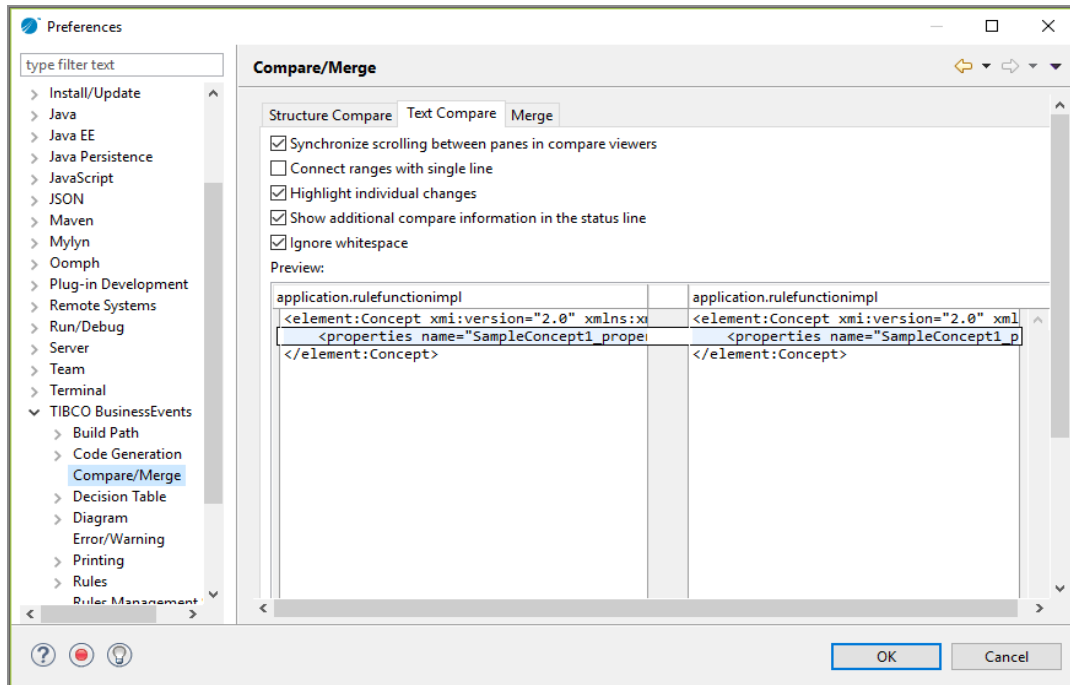
Compare/Merge Preferences

This tabbed dialog is in the TIBCO BusinessEvents preferences. It applies to decision tables.

In the Structure Compare tab, set preferences for initial settings and colors. The effect of your choices is shown in the lower panel.



In the Text Compare tab, you can set preferences for text. The effect of your choices is shown in the lower panel.



The Merge tab has one option: Allow columns to be automatically merged.

Rules Management Server Preferences

In this group of preferences, you can set the following:

- Authentication URL history size
- Checkout URL history size

Working with RMS Projects and Artifacts in Decision Manager

The projects and artifacts that were added to RMS can be used in Decision Manager to create business rules.

Starting and Logging in to RMS

Before you can work on the RMS projects, you need to start and log in to the RMS.

Procedure

1. Start the RMS server using one of the following ways:
 - Execute `BE_HOME/rms/bin/be-rms.exe` (or `be-rms.sh`, depending on your operating system), with valid arguments. For example, on Windows you open a command window in `BE_HOME/rms/bin` and execute: `be-rms.exe`
 - (Windows only) Select **Start > All Programs > TIBCO > TIBCO_HOME > TIBCO BusinessEvents_{version} > Start Rules Management Server**.

The server is successfully started when the following message is displayed in the command window:

```
Info [HTTP-Channel-Startup] - [driver.http] Channel server for
HTTP Channel [Port:5000] successfully started
```

2. In Decision Manager, from the RMS menu, select **Login**.

You see the **Login** dialog.



Tip: RMS menu options are only available if there is a connection to the server. (Before logging in, only the **Login** option is available.) If the connection fails, all options are dimmed. Upon reconnection, the **Login** option is available again.

3. To select a different RMS server, enter its URL or select the correct URL from the Management Server URL drop-down list. Ask an administrator for the RMS server URL if you do not know it and it is not in the list already.
4. Type the username and password.
The credentials are associated with the user role. As shipped the credentials are admin/admin.
5. If you want to save the password, select the **Save Password** check box. This means that on your subsequent logins, the password field will be populated automatically.
6. Click **OK**.
Credentials are verified and if log in is successful you see the message “You have logged in successfully.”

Result

In TIBCO BusinessEvents Studio, from the RMS menu, select **Logout**. You see a success message. All RMS menu options are dimmed except **Login**.

Checking Out a Project

If RMS is used in your environment, then you must log into RMS and check out a project from RMS before you can work with the project resources in the Decision Manager UI.

If you want to generate a deployable EAR file, ensure that you select all or at least all required artifacts. The EAR file may not build or deploy correctly if required artifacts are missing.

Depending on your role or roles, you may not have permission to check out all resources or to do all of the documented tasks.

For instructions on updating a project you already checked out, see [Updating \(Synchronizing\) a Project](#).

Procedure

1. Ensure no local projects are selected. The RMS checkout options are not available if a local project is selected.

2. From the RMS menu, select **Checkout**.

The **Checkout** dialog displays.

3. To select a different RMS server, enter its URL or select the correct URL from the **Management Server URL** drop-down list.

Ask an administrator for the RMS server URL if you do not know it and it is not in the list already.

4. In the **Projects List** section, click the project of your choice.

The list displays RMS projects stored in the base location for this server. The project name displays in the **Project Name** box.

5. Click **Get Artifacts**.

A list of the artifacts (resources) from that project appear below the **Project Name** box. The artifacts include all the resources from the TIBCO BusinessEvents Studio project, such as rules, rule functions, virtual rule functions, decision tables, concepts, events, domain models, shared resources, channels, global variables, and so on.

If you are checking out an artifact for the first time, the ChangeType for that artifact is ADD. For subsequent check outs, the ChangeType is MODIFY.

6. Select the artifacts you want to work on locally and click **OK**.

The **Checkout** dialog displays the list of artifacts you have selected for checking out.

7. Click **Finish**.

The selected artifacts appear in BusinessEvents Studio Explorer.

Updating (Synchronizing) a Project

It is recommended that you log into the RMS server, and update (synchronize) your local copy of a project with the most recent changes before continuing to work on it.

Other users may have checked in changes, additions, and deletions that are now available for checkout. You can choose what updates to accept. Changes and additions to individual decision tables and the domain model are listed.

See [Comparing and Merging Decision Tables](#) for more information on comparing decision tables and merging changes.

Procedure

1. In Decision Manager, do one of the following:

- From the **RMS** menu, select **Update**.
- Right-click the project, and select **RMS > Update**.

The **Update** dialog displays.

2. If you did not already select a project, select one from the **Projects List** section, and click the **Get Artifacts** option.

The artifacts from that project are listed below the Project Name box. The artifacts include rules, rule functions, virtual rule functions, decision tables, and domain models.

The ChangeType of each artifact is shown as ADD or MODIFY.

3. Check the check boxes of the artifacts you wish to update. You can also use the **Select / deselect all** check box at the bottom to check or uncheck all the listed check boxes.

4. Click **OK**.

The Update... Successful dialog is displayed.

5. Refresh the project either by right-clicking the project folder and selecting **Refresh**, or by pressing F5.

Committing Artifacts

Every update to the project needs to be committed to the RMS. These commits then follow the approval process before being applied to the project in RMS.

Copies of submitted artifacts are saved in cache, with the names of the users who submitted them, version numbers, and the status **Committed**. A task is created for the Approver role to review the check-in requests.

A Request ID is assigned to each commit request. This ID is used in other dialogs to allow you to view details on the request.

You can view the entire audit record of previous status changes for a check-in. This is especially useful for multi-stage approval process so that different reviewers actually share views on why a particular check-in was approved or rejected.

Procedure

1. Select a **Project**, and do one of the following:

- From the **RMS** menu, select **Commit**.
- Right-click the project, and select **RMS > > Commit**.

The **Commit** dialog lists the newly added, modified, and deleted (red) resources. If the project has no changes to commit, the **OK** button is disabled.

2. In the Comments panel, type any comments you want the Approver to read.
3. Select the artifacts, such as rules, rule functions, decision tables, or domains you want to commit.

Optionally, you can select all the artifacts by checking **Select / deselect all**.

4. Click **OK**.

Similar to any repository, a record of each committed item is displayed with a check-in ID.

Committing Project Artifacts to RMS at the Command Line

You can commit various project artifacts to RMS using command line as well so that TIBCO BusinessEvents Decision Manager users can check them out to work locally on them.

i Note: The Studio Tools utility runs only on Microsoft Windows and Linux platforms as it has dependencies on Eclipse.

Procedure

1. Navigate to `BE_HOME/studio/bin/` and open a command prompt.

2. Execute a command with this format (all one line) at a command prompt:

```
studio-tools -rms commit [-h] -studioProjPath <Studio Project Workspace Path>
-rmsBaseURL baseURLForRMS -rmsUsername RMSLoginUsername [-rmsPassword
RMSLoginPassword] -artifactPath projectPathtoArtifact -artifactType
{decisiontable | domain | channel | concept | event | timeevent |
metric}
```

For example:

```
studio-tools -rms commit -studioProjPath
D:\Workspace\CreditCardApplication -rmsBaseURL
http://sesame.acme.com:5000/Channels/MyChannel/ -rmsUsername admin
-rmsPassword admin -artifactPath \Virtual_RF\BankUser -artifactType
decisiontable
```

Result

The following table provides detailed information about the options.

TIBCO BusinessEvents Studio Tools Options for RMS Commit

Option	Description
-rms commit	Within the rms category of operations, specifies the commit tool for committing project artifacts to RMS.
-h	Optional. Displays help.
-studioProjPath	File path to the TIBCO BusinessEvents Studio project that contains the artifacts to be committed.
-rmsBaseURL	Base URL of the RMS server.
-rmsUsername	Username required to log into the RMS server

Option	Description
<code>-rmsPassword</code>	Password, if required to log into the RMS server.
<code>-artifactPath</code>	TIBCO BusinessEvents Studio project path to the artifacts to be imported. Valid values are decisiontable, domain, channel, concept, event, timeevent, ruletemplate, ruletemplateview, and metric. When another user checks out the project, the committed artifacts are copied to that user's project.

Viewing the History of RMS Artifacts

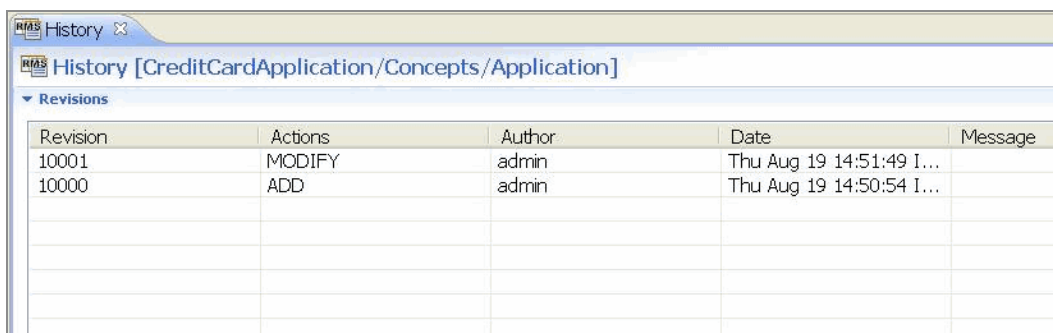
The History View shows the history of all the artifacts checked in to RMS and allows you to compare the revisions.

It displays all the revisions of the resources, which are previously checked-in or committed.

Procedure

1. Select any single artifact in the Studio Explorer, and then go to **RMS > Show History**. All revisions related to the resource are displayed in the History View as fetched from RMS.

Result



The screenshot shows a window titled 'History' with a sub-header 'History [CreditCardApplication/Concepts/Application]'. Below this is a section labeled 'Revisions' containing a table with the following data:

Revision	Actions	Author	Date	Message
10001	MODIFY	admin	Thu Aug 19 14:51:49 I...	
10000	ADD	admin	Thu Aug 19 14:50:54 I...	

The History View Reference

The History View consists of two sections: the Revisions section, which list all revisions of the artifact, and the Details section, which lists detailed information about the selected revision.

Revision Section

The Revision Section Columns

Property	Description
Revision ID	The unique ID assigned to the revision
Actions	Action performed on the resource. The values are: • ADD • MODIFY • DELETE
Author	The user who performed the action on the resource.
Date	The date and time when an action was performed on the resource.
Message	Comment entered during committing the resource for the revision.

The Details section

The Details section displays the resource level information on selection of one or more revisions in the Revision section.

The screenshot shows the 'History' view for the artifact 'CreditCardApplication/Concepts/Application'. It is divided into two sections: 'Revisions' and 'Details'.

Revisions Table:

Revision	Actions	Author	Date	Message
10001	MODIFY	admin	Thu Aug 19 14:51:49 I...	
10000	ADD	admin	Thu Aug 19 14:50:54 I...	

Details Table:

Action	Path	ApprovalStatus
ADD	/Rules/RuleSet/Add_New_Bankuser_Rule	PENDING_APPROVAL
ADD	/Concepts/Card	PENDING_APPROVAL
ADD	/Channels/RV	PENDING_APPROVAL
ADD	/Concepts/Application	PENDING_APPROVAL
ADD	/Virtual_RF/ProcessApplication_VirtualRuleFunction	PENDING_APPROVAL
ADD	/Events/ApplicationReceived	PENDING_APPROVAL
ADD	/Virtual_RF/Application_VirtualRuleFunction	PENDING_APPROVAL

The Details Section Columns

Property	Description
Action	Action performed on the resource. The values are: • ADD • MODIFY • DELETE
Path	Path of the resource in the project.
ApprovalStatus	Approval status for the resource in selected revision.

Comparing and Merging Artifacts

View the difference between two copies of artifact and merge those copies to sync any changes done by others, or revert to any previous revision.

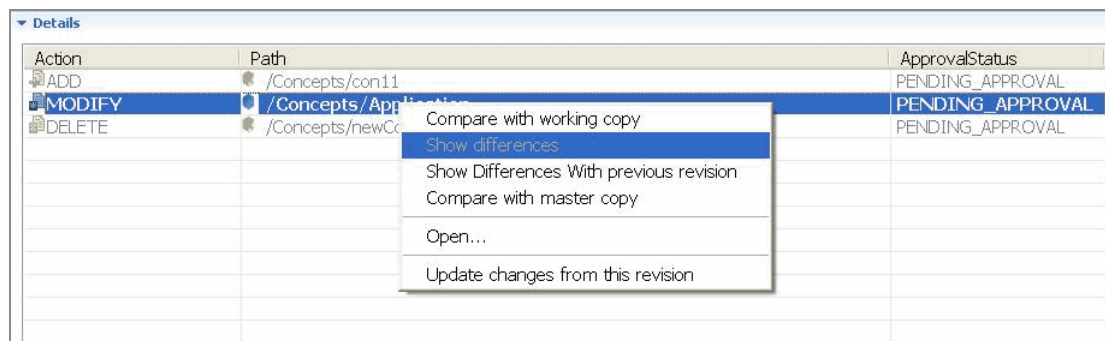
The History View provides options to compare between:

- Two revisions of the same artifact
- One revision and one working (local) copy of an artifact
- One revision and the master or approved copy of an artifact

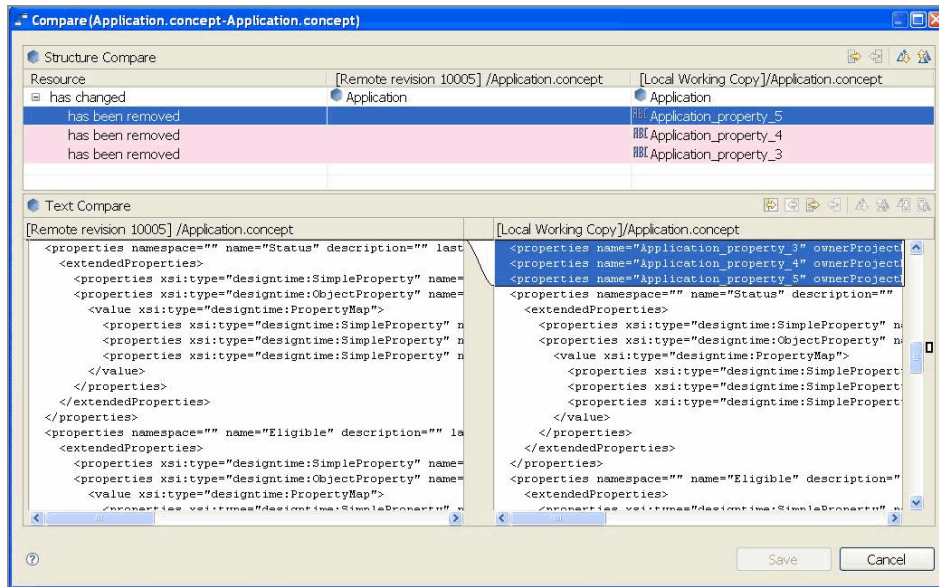
In addition, the History View also lets you merge structure or text into a local copy from any remote copy. Structure compare is available for all RMS artifacts except for Rules and Rule Functions. Text compare is available for all RMS artifacts. You can also revert the changes you merged in the local copy.

Procedure

1. Right-click on any artifact in the Details section of the history view.
 - Select **Compare with working copy** to compare the RMS copy with the local copy.
 - Select **Show differences** to compare between two revisions of the same artifact.
 - Select **Show difference with previous revision** to compare the latest copy with the most recent version of the artifact.
 - Select **Compare with master copy** to compare the local copy with the approved copy of the artifact.



2. In the **Compare** dialog, select the appropriate merge option, such as **Copy from left to right** or **Copy All from right to left**, and so on.



3. Click **Save**.

The changes are reflected in the local copy. For Remote copy comparison, merging operations are disabled.

Reverting to a Master Copy

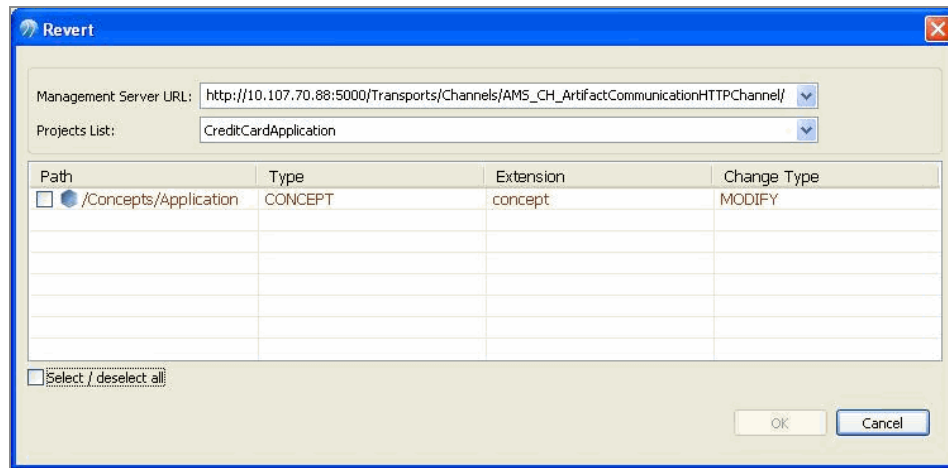
When you revert the changes done to an artifact, the working (local) copy of the artifact gets overwritten by the master copy.

Note: You can revert changes done to the working copy of a single resource.

Procedure

1. In Studio Explorer, do one of the following:
 - Right-click an artifact, and select **RMS > Revert**.
 - Select an artifact, and click **Revert** from the **RMS** menu.

The Revert dialog appears.



2. Select the project of the artifact in the **Projects List**.
3. Check the check box of the artifact you want to revert.
4. Click **OK**.

Rule Building with Decision Tables

As a business user, you check out projects from RMS, build decision tables, and submit them for approval.

Decision tables provide a graphical way to build complex business rules. You create table columns by dragging and dropping predefined properties onto the decision table framework. The properties belong to ontology resources defined in the TIBCO BusinessEvents project. You can only use the properties specified in the VRF; however, Columns can be created in other ways too. You then define threshold values (conditions) and actions in the cells of the table. Each row can be thought of as one rule in a table made up of many rules. The individual rules are often straightforward, as in the following examples.

- Three Rule Conditions:

```
Person.age < Max(20, Parent.age)
Person.creditscore >= Math.function(...)
Person.gender == "female"
```

- Three Rule Actions:

```
Application.status = "ACCEPTED"
Application.credit = 4000
sendNotification()
```

However, one decision table can consist of hundreds, even thousands of rules each of which is executed only when its specific conditions are satisfied.

Exception Tables

Each decision table can optionally have another table known as an exception table. The purpose of the exception table is organizational: it enables you to separate the business logic of the main decision table (added by business users) from any non-business logic (generally added by technical users). For example, in the exception table, you could capture situations where fields are blank or contain invalid values, and define actions that send notifications or set return values. The rows of a decision table plus the rows of its

exception table are considered in an RTC. If you prefer, you can put non-business logic in the main table instead of using an exception table.

Table Analyzer

The Table Analyzer feature analyzes decision tables and reports problems, such as uncovered ranges for conditions, uncovered domain entries, different set of actions for identical conditions. Table Analyzer creates a sparse matrix data structure representing an optimized form of decision tables in memory.

Working with Decision Table

In Decision Manager you can create a decision table, perform various operations on it, and set its properties as per your requirement.

 **Note:** To change the appearance of the decision tables, select **Window > Preferences > TIBCO BusinessEvents > Decision Table > Appearance**. See [Decision Table Related Preferences](#) for more details.

The maximum size of a decision table is 64 KB.

Adding a Decision Table

A decision table can only be added to a virtual rule function.

For details on adding a VRF, see *TIBCO BusinessEvents Developer's Guide* of the TIBCO BusinessEvents documentation set.

Procedure

1. Right-click in Studio Explorer, and select **New > Decision Table**.
You see the **New Decision Table** Wizard.
2. If you right-click on a VRF, that VRF gets selected in the **New Decision Table** Wizard. If you want to add this decision table to a different rule function, select it in the **Select Virtual Rule Function** area.

3. In the **Decision Table Name** Field, type a name for the decision table.
4. Click **Finish**.

You see the decision table ready for work.

Setting Decision Table Properties

You can determine many aspects of decision table execution using its properties.

Procedure

1. In the **Effective Date** field, you can optionally define the date and time on which the decision table becomes valid in the runtime application. Click the calendar icon and use the Calendar dialog to set the date and (as desired) the time.
2. Similarly in the **Expiration Date** field, you can optionally define the date and time after which the decision table is no longer valid in the runtime application.
3. Check the **Take Actions of One Row at Most** check box if you want the decision table to stop after one row (rule) passes the condition tests. Only the actions of that one row are taken (until the next time the decision table's rule function is called).
4. In the **Priority** field, set a priority as desired. When a VRF has multiple implementations (decision tables), the order in which the decision tables execute can be defined using the decision table's Priority setting. (You can also set row priorities. See [Setting Rule \(Row\) Priorities](#) for details.)

Viewing Decision Trees

A decision tree is a visual representation of a decision table. The condition/action columns in the table are the nodes in the decision tree.

It is a read-only representation, in which you can re-arrange the edges and the nodes, but you cannot edit the decision tables from a decision tree.

Procedure

1. In Studio Explorer, right-click a decision table, and select **Create Decision Tree**.

The decision tree appears in the editor.

2. If a decision table is updated, you can click the **Refresh** button in the toolbar to update the tree.
3. You can do the following by right-clicking on the canvas in the editor:

Option	Description
Simplify Tree	If you click this, you need to specify visible levels from the root to be displayed. The decision tree appears in the simplified form.
Unfold Entire Tree	This displays the simplified tree unfolded to the last visible level from the root.

4. You can also select the following options by right-clicking a node in the decision tree:
 - **Fold One Level**
 - **Fold N Levels...**
 - **Fold All Levels**
 - **Fold One Parent Level**
 - **Fold N Parent Levels...**
 - **Fold All Parent Levels**
 - **Unfold Selected**
 - **Unfold**

Deleting Decision Tables

If the table you want to delete has been checked into the RMS project, then you must submit the deletion for approval.

After approval, the deleted table is removed from the RMS project, but it is not removed from local Decision Manager projects. When an artifact deletion is approved, the artifact is permanently deleted from the RMS project. However, When Decision Manager or Studio users update their projects, the deleted artifacts do not appear in the list of changes.



Warning: Users who have already checked out the project must manually delete their local copy of the artifact.



Tip: If you are not sure whether a table you deleted was committed to the RMS project in a prior action, select **Project > Commit**. If the deleted table does not appear in the Changes Made panel, then it is a local table.

Procedure

1. Do one of the following:
 - In Studio Explorer, right-click the decision table name and select **Delete**.
 - In Studio Explorer, select the decision table name. Then select **File > Delete**.
2. If the table has been committed to the RMS project, choose one of the following options as appropriate:
 - Select **Project > Commit** so that the RMS master copy of the table is also deleted. OR
 - Select **Project > Update** to replace the local deleted copy with the RMS project version of the table.

Setting the Decision Table Effective Period

You can set effective and expiration dates to define when a table is active in the runtime TIBCO BusinessEvents application.

When you specify a date, you also specify a time, in hours, minutes, and seconds. If no effective or expiration dates are specified, the decision table is executed every time it is invoked.

You can set an effective date without setting an expiration date. Such a table takes effect on the date and time specified and never expires.

If you want to set an expiration date, however, you must also set an effective date. Such a table becomes active on the effective date and time specified, and remains active until the expiration date and time specified.

Procedure

1. Open the decision table in the decision table editor.
2. Select the **Properties** view for the table.
3. Click the **Calendar** button to the right of the **Effective Date** field and select the date and time as desired.

To set the time, highlight the hours or minutes or seconds and use the spinner to set the value.

To clear a value from the field, click the red **X** button.
4. To set a value in the **Expiration Date** field, use the same techniques as in the previous step.
5. Click **Apply**.

Comparing and Merging Decision Tables

You can compare a decision table with another decision table in the project. You can also copy all or selected differences between the two tables under comparison.

i Note: To change the appearance of the Compare view, select **Window > Preferences > TIBCO BusinessEvents > Compare/Merge**.

From the structural compare window, you can merge changes in a decision table.

i Note: You cannot merge remote decision table versions. The Copy actions are disabled when remote versions are compared.

Procedure

1. In Project Explorer, select two tables you want to compare.
2. Right-click and select **Compare With > Each other** from the context menu.

You can also use buttons to perform the compare and merge operation (see [Compare Editor Buttons](#)).

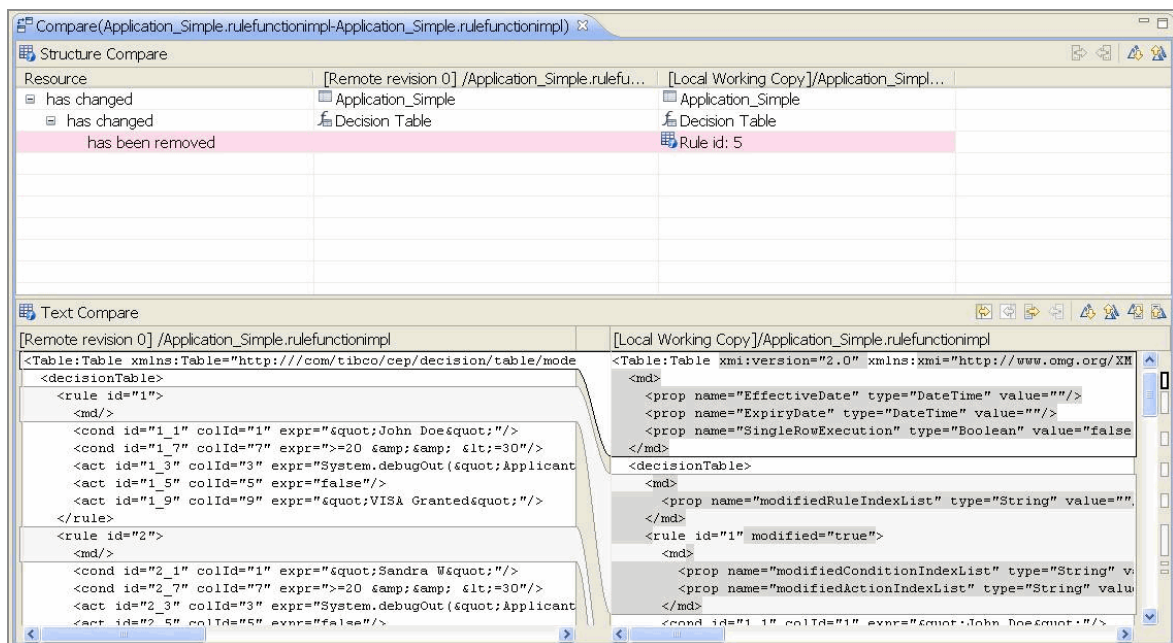
The **Compare** editor displays the two tables.

- To merge (copy), select individual differences and click one of the copy actions.

For example, to overwrite a local change with the remote (approved) version, select the change and click **Copy right to left**. In the left column, the entry for the difference disappears.

This feature can be useful when the approved version of the decision table is newer than the local version, but you do not want to overwrite all changes in the local version (using the **Update** command).

It can also be useful to copy useful parts of one local decision table to another local decision table.



Compare Editor Buttons

The compare editor buttons provides alternate way to compare and merge two decision tables.

Button	Description
	Compare tables.

Button	Description
	Next difference. Previous difference.
	Next change. Previous change.
	Copy from left to right. Copy from right to left.
	Refresh the history of the current item (History view).
	Update local copy with this revision (History view).

Columns and Rows (Rules) in a Decision Table

A decision table rule is a row in the decision table. It has one or more conditions and one or more actions.

Each condition cell is equivalent to one condition (one line) in the TIBCO BusinessEvents rule editor condition area. Similarly each action cell is equivalent to one action (one line) in the TIBCO BusinessEvents rule editor action area (that is, one rule). Decision table rules are like business rules. The rule that calls the virtual rule function that implements the decision table participates in inferencing in the usual way. When the virtual rule function is called, the decision table rules are applied.

Conditions and Actions

The columns of a decision table are made up of condition columns on the left, and action columns on the right. Each column represents one condition or one action.

A condition is a test that must evaluate to true before the action is executed. If a decision table rule uses multiple conditions, all conditions for a row must evaluate to true in order for the action to execute.

In each row (rule) you define the specific conditions and actions. For example, if a condition column is Age (using a concept property of that name), then each row can define

a different age range. The action for each row would define what action to take if a given concept instance contains an age property within the specified range.

If you add a second condition column called Income, then before the action is taken, a concept instance would be tested to see if both the age and the income are within the ranges specified in the rule's conditions.

i Note: Conditions that are blank, contain an asterisk, or are disabled are ignored, and are treated as if they evaluate to true.

Regular and Custom Conditions and Actions

Regular condition

A regular condition is the value of an entity specified in the VRF scope, or a simple comparison with the value - is greater than, is less than, is greater than or equal to, is less than or equal to.

Custom condition

A custom condition can use the rule language, standard functions, and data in the scope of the function at runtime (for example, scorecards and global variables). It can contain complex formulas.

Regular action

A regular action sets the value of an entity specified in the VRF scope.

Custom action

A custom action can use the rule language, standard functions, and data in the scope of the function at runtime to do whatever is desired. For example, the action could be to send an event out to a different system for follow-up.

Use of Non-literal Values in a Regular Condition or Regular Action

Suppose event A and event B are in scope, and event A has property ZZ, and event B has property YY. Both properties belong to the same data type. Suppose you then drag property ZZ to a condition or action column. In the cell, you can then specify a value as b.YY. The effect is different depending on the type of column:

- In a condition column, this means: compare the value of property ZZ with the value of property YY.
- In an action column this means: set the property ZZ to the value of YY.

Working With Decision Table Rules (Rows) and Columns

To manage and edit a decision table, Decision Manager provides various operations that can be performed on the rows and columns.

Creating Variables and Assigning to Them

You can create variables and assign to them in a rule function that is called from a condition. (Note that you cannot do this in a regular TIBCO BusinessEvents rule condition.)

Undo Operations on Edits

The following actions can be reverted by clicking **Undo** when working on decision tables:

- Dragging a column into a decision table
- Removing a column from a decision table
- Adding a new row to a decision table
- Removing an existing rule
- Duplicating rules
- Modifying condition/action cells

You cannot perform an Undo operation immediately after a Save operation.

Aliases in Table Cells Requires Parentheses

To use an alias in a table cell, enclose it in parentheses. For example if `cust` is the alias for the Customer concept, you would reference the `Customer.address` property as follows: `(cust.address)`. If you omit the parentheses, the EAR file will not build.

Adding Decision Table Condition and Action Columns

You can add any properties that are used as arguments as conditions or actions.

However you cannot use attributes. Event and concept attributes (such as `extId`) are not visible. You cannot use event payloads, which are attributes. You can also create custom columns that, for example, perform comparisons between values.

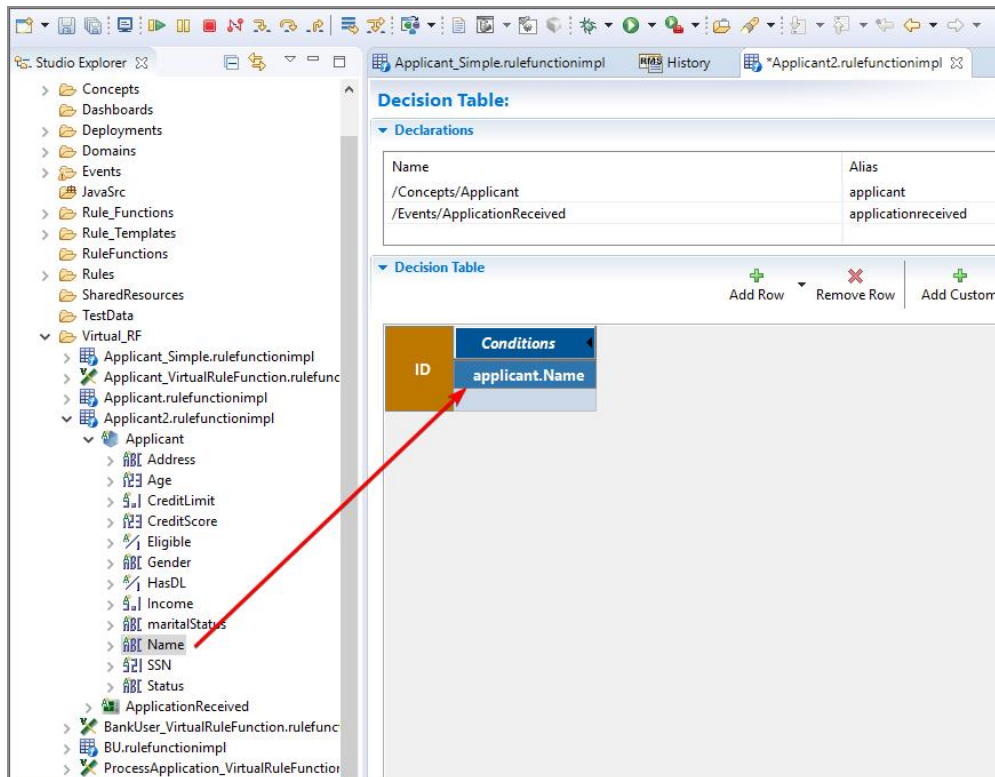
Procedure

1. Drag a property from the Argument list below a decision table to the decision table editor.

The first argument drag to the decision table editor is always added as Condition.

2. After adding the first condition, do one of the following:

- Drag the argument to the existing condition column to add another condition.
- Drag the argument outside the condition column (empty space in the decision table editor) to add action column.
- You can add new custom conditions and actions using the **Custom** button in the Decision Table editor.



Note: When you add columns in a decision table, in the tree, the properties are populated just below the table you create. Expand the table name, and drag and drop properties to the condition or action column.

Defining Custom Columns Using Substitution

You can specify complex entries in a column by using column substitution. This feature works where all the entries in a column follow a specific pattern.

As an example, suppose all entries in a column use a comparison such as this, but with different values in each case:

```
Math.maxInt(MyAlias.Value, MyAlias.MaxValue) < 50
```

Instead of typing such entries in each cell, you could use substitution.

Procedure

1. In the column header, specify (right-click the column and select **Field Settings**) the pattern: `Math.maxInt({0}, {1}) < {2}`
2. In each of the column cells, specify parameter values, using a semicolon delimiter:
`MyAlias.Value; MyAlias.MaxValue; 50`
3. Press Enter so that the cell contents are substituted with the pattern of the column name.

Arrays in Columns

You can create columns that use arrays, including nested (that is, contained or referenced) concept arrays and properties.

The following figure displays the decision table that uses arrays in columns:

The screenshot shows a software interface for a decision table. At the top, there's a header bar with the text "/Concepts/ContainerConcept", "containerconcept", and "false". Below this is a toolbar with icons for "Add", "Remove", "Custom", "Fit Content", "Merge Rows", "Duplicate Rule", "Show Text", and a "Search" button. A text box contains the formula: `System.debugOut("Value [0:0:0]" + containerconcept.prop_con[0].prop_con[0].prop_double[0])`. Below the toolbar is a table with three columns: "ID", "Condition Area", and "Action Area". The "Condition Area" column contains the text `containerconcept.prop_con[],prop_con[],prop_double[]`. The "Action Area" column contains the text `containerconcept.prop_con[],prop_con[],prop_double[]` and "Custom Action 1". The first row of the table has the ID "1", the condition `[0:0:0]0`, and the action `[0:0:0]000` followed by `System.debugOut("Value`.

ID	Condition Area	Action Area
1	<code>[0:0:0]0</code>	<code>[0:0:0]000</code> <code>System.debugOut("Value</code>

Defining Columns

You can drag and drop any level of nested (contained or referenced) concept arrays, and properties of such arrays, to the Conditions or Actions areas to create a named column. You can specify the array index against which to check a value (condition), or to set a value (action).

For example, suppose a concept A has a property B which is an array of contained concept B, and concept B has a property `intProp`. If you drag `intProp` into the decision table to create a column, the new column name is as follows: `a.b[].intProp`

Specifying Values

When the expression in the column name contains the access to a position within an array (as in the example `a.b[].intProp`), then you can specify which position must be used in the cell of a table row.

To specify the position, enclose the position index within square brackets, followed by the value. For example, to specify a value of 100 for a specified index in a cell, you might enter the following:

```
[0]100
```

For an action, this is the equivalent of specifying the following: `a.b[0].intProp = 100`. For a condition it would be: `a.b[0].intProp == 100`.

Omitting the Array Index

If you omit the array index and simply set the cell to a value such as 100, by default the array component's last element is used. This would be the equivalent of specifying `a.b[a.b@length - 1] = 100` for an action, and `a.b[a.b@length - 1] == 100` for a condition.



Note:

- To avoid ambiguity you must specify the array index in the following cases:
 - Multiple loop indexes
 - String expressions
- If you omit the array index and no value is present at the last element, an exception results. It is recommended that you specify the index, and ensure that all indexes specified have values.

Using Function Evaluations

You can also specify index values using a function evaluation, for example:

```
[Number.intValue("0",10)]10
```

Using Multiple Loop Indexes

If multiple loop indexes are involved, as in the case of nested concepts, use the colon character (:) as the delimiter for indexes. Always specify the index, to avoid ambiguity. For example, given this nested concept situation:

```
a.b[].c[].d[].intProp
```

You can also specify a condition cell value as follows:

```
[0:1:System.getGlobalVariableAsInt("GlobalVariableName")]>=100
```

Which is the equivalent of the following:

```
a.b[0].c[1].d[System.getGlobalVariableAsInt("GlobalVariableName")] >= 100
```

Moving Resizing and Sorting Columns

Decision Manager provides function to perform certain operations on columns that would help in working on decision table.

- Rearrange the order of columns within the **Condition** area and within the **Action** area. To rearrange columns, right-click the column you want to move and select an option:
 - **Move to the beginning**
 - **Move to the left**
 - **Move to the right**
 - **Move to the end**
- Drag the boundaries of columns to manually resize them.
- The **Fit Content** option resizes the columns to fit the content in cells.
- Sort the columns by clicking their headers, and selecting the appropriate sorting order.

Setting Default Value in Decision Table Column

Using the **Field Settings** option of the column, you can set a default expression for the column for all new rules.

Procedure

1. Right-click any one of the column headers to open the column's option menu.
2. Select the **Field Settings** option.

The Field Settings dialog is displayed.

3. Enter the default value for the column in the **Default Expr** text box.

This default value is used for the column, whenever you add a new row.

The **Include existing rules** check box is activated.

4. Select the **Include existing rules** check box to fill all the blank cells for the column with the default value specified in the **Default Expr** text box.

5. Click **OK**.

The Field Settings dialog closes.

Result

Now the value of the **Default Expr** field is used as the default value for the column for new rules. If you have selected the **Include existing rules** check box, all empty cells for the column in the decision table are replaced by the default value.

Deleting Rows and Columns

You can remove the rule as well as any condition or action that you do not require.

You cannot remove the last condition column because a minimum of one condition column must exist if you have any action columns.

When you remove columns, certain rows may also get merged because they now have the same set of conditions. If this happens, if you have non-custom actions and the rows have the same priority, the last row to be merged has its actions merged and the others deleted.

- To delete one or more rows, select the rows and click **Remove**.

i Note: The deleted row IDs are not reused when you add more rows.

- To delete one or more columns, right-click the column header and select **Remove**

Duplicating Rules (Rows)

When you require to add new rows which have almost same values as the existing rows, only with the minor changes, the duplicate function is very helpful.

Procedure

1. Click **Duplicate Rule** to duplicate a row.

This adds a new row with the same conditions and actions, but with a different ID. You can then edit the row as needed.

Merging Rows

In a large decision table, the merge function helps to merge any existing duplicate condition and thus makes the decision table more efficient.

Procedure

1. Click **Merge Rows** to merge rows with the same conditions into a single row data. Actions for the first row are used for the merged row.

The Merge Rows button acts as a toggle: to toggle between merged and unmerged rows, click the **Merge Rows** button again.

i Note: The initial state of the merge rows button is determined by a decision table preference. To set the initial state, select **Window > Preferences > Decision Table** and set the property **Automatically merge rows** as desired. See [Decision Table Related Preferences](#)

Setting Rule (Row) Priorities

Each row is like a separate business rule. You can control the order in which sets of rows are executed using the row priority setting.

Rows with higher priorities are executed before those with lower priorities, as follows:

1. First, all conditions are checked for all rows that have the highest priority. (The checking order within a set of rows with the same priority is not determinate.)
2. Then the rule actions for all of those rows whose conditions evaluate to true are executed. (The execution order is not determinate. The runtime engine optimizes rule execution.)
3. The process is repeated for all rows with the next highest priority, and so on.

Ten is the lowest priority and one is the highest. Five is the default priority.

Procedure

1. Select the row whose priority you want to set.
2. Select the **Properties** view, and select the **Rule** side-tab.
3. In the **Priority** field, select the appropriate value and click **Apply**.

Filtering Rows

When you are working with large tables, it can be helpful to filter the rows so you can focus on certain ones.

You can filter on multiple columns. For example, you might filter column A to show only rows with one value for A, and another column B to show only rows with one value for B.



Note: To enable or disable filtering on conditions or actions or both, select **Window > Preferences > Decision Table** and set the Condition field filter and Action field filter settings as desired. See [Decision Table Related Preferences](#)

Procedure

1. Open the decision table in the decision table editor.

2. Click the down arrow to the right of a column header.

You see a list of all values that have been used in the column's cells.

3. Do any of the following as appropriate:

- Clear the **All** check box to clear all the other check boxes, then select the check boxes beside the value or values you are interested in.
- Clear the check box for values you are not interested in.

4. Click **OK**.

Only rows containing the checked values display.

5. Repeat to filter on additional columns as desired.

What to do next

To remove the filter conditions when you are done, select **All** in each of the filter columns.

Domain Models in Table Cells

A domain model specifies the values that you may find useful for defining ontology item properties.

For example, instead of typing text for a certain concept property, you can pick a value from a list, or enter a value within a predefined range. Domain models are defined in the TIBCO BusinessEvents Studio project, not in TIBCO BusinessEvents Decision Manager itself.

All domain model values can have optional descriptions. A preference determines whether domain model values or their descriptions appear in decision table cells. For some applications displaying descriptions can make the table easier to understand. For example suppose the value is a code such as 23, and the description is North West. Users will find it is easier to work with the description than the code. As another example, for a Boolean data type, the description can provide words such as Accepted and Rejected for the values True or False. At runtime, the actual value is used.

Catalog Functions in Table Cells

You can also use the existing catalog functions in the table cells to perform some standard functions.

To use a catalog function in a table cell, first expand the catalog functions, which are collapsed on the right side of the window. Select a function. Each function has helpful tooltips. Drag and drop the function to the function area (marked by the ) or to the cell you are editing.

For example, if you want to compare some attribute that is of type integer against the rounded value of 39.99, you could specify a condition `< Math.round(39.99)` by dragging and dropping the Math round function from the Standard Functions panel and then entering the arguments of that function.

Setting the Date-Time Values Using the Calendar Dialog

The Calendar dialog in Date-Time Cells enables you to set date time values. You can also set values manually, and you can use comparison operators with dates.

Decision Manager has two types of calendar dialogs. One is a simple calendar, used to set effective or expiry date selection in the general properties for a decision table. The other is the advanced calendar, available in date-time cells. It offers options such as matches, before, after, between and so on, that enable you to analyze and choose an exact date time, by comparing with the previous date.

i Note: When working with date literals in cells, you can use operators such as greater than (>), less than, (<), greater than or equal to (>=), less than or equal to (<=) and Conditional-Or (| |). For example:

```
> 2010-11-15T00:50:06
```

Procedure

1. Click the calendar icon, to open the Calendar dialog.
2. Select the date and time in the right pane.
3. Click **OK**.

Supported Operators

TIBCO BusinessEvents Decision Manager supports several operators for Decision Tables that you can use in the expressions.

Named or Scoped Conditions must Evaluate to Boolean

For logical operators, named conditions will have to use them explicitly from the second operand onwards.

For example, if the LHS of the condition is the column name, `bankuser.Age`, the following are valid entries:

```
50 || ==60 , 40 && ==80, 20 || >=80
```

Custom Conditions must Evaluate to Boolean

Same as earlier specified condition, except that you must specify the entire Boolean expression in the cell, including LHS.

Actions or Custom Actions

Named actions perform assignment of the cell expression to the column name property.

Custom actions are treated as normal strings and can be anything that you would enter in the THEN section of rules (LHS included).

Supported Operators

Operator	Description
==	Equals to
!=	Does not equal to
<>	Does not equal to
>	Greater than

Operator	Description
>=	Greater than or equals to
<	Less than
<=	Less than or equals to
	Logical OR
&&	Logical AND
+	Addition or String concatenation (depending on context)
!	Logical NOT (only with boolean)
-	Subtraction
*	“Don’t Care” (that is, ignore), or multiplication (depending on context)
/	Division
=	Assignment
.	Scope resolution
()	Operator precedence order or function call (depending on context)
[]	Array declaration or array indexing (depending on context)
{}	Block resolution or array Initialization (depending on context)
++	Increment (used only with custom actions)
--	Decrement (used only with custom actions)

Exporting and Importing Decision Tables

You can export and import decision tables between TIBCO BusinessEvents Decision Manager and Microsoft Excel spreadsheets.

You can merge Excel data into an existing decision table, or you can overwrite an existing table.

If you are used to working with Microsoft Excel, you may prefer to build decision tables in that user interface and then import the table into Decision Manager. To get the basic format, you can set up a simple table in Decision Manager then export it to Excel and develop the content there.



Note: If your table has custom conditions or custom actions, use the Excel spreadsheet header names "CustomCondition" or "CustomAction" as appropriate.

Exporting a Decision Table to a Microsoft Excel File

A table export includes a decision table and its exception table in one spreadsheet.

If you export domain models associated with properties used in the decision table, the domain models appear on a separate sheet. You can add more validations with ranges or dependency lists, and so on.

The excel sheet validates the column values such that you can use only the drop-down values coming from the union of all the values in the domain models. You cannot add any value outside the domain models.

If the preference is set, the Column aliases are exported. To set the preference, go to **Window > Preferences > TIBCO BusinessEvents > Decision Table**, and set the Export column alias option as desired. See [Decision Table Related Preferences](#)

i Note: The rows with the priority setting of 5 in a decision table row are not exported(because 5 is the default value).

Procedure

1. Select the decision table you want to export.
2. Perform one of the following:
 - From the **File** menu select **Export**.
 - Right-click the decision table and select **Export**.The **Export** dialog is displayed.
3. Navigate to the directory in which you want to save the file and provide a file name.
4. Click **OK**.

Exporting Decision Tables to the Microsoft Excel Files at the Command Line

You can export a single decision table or all the decision tables from a project folder to the respective Microsoft Excel files using the command line.

Procedure

1. Navigate to *BE_HOME/studio/bin/* and open a command prompt.

2. Execute a command with the following format (all in one line) at a command prompt:

```
studio-tools -dt exportToExcel [-h] -studioProjPath studioProjectDir -
dtPath decisionTablePath -excelPath exportedExcelFilePath [-useColumnAlias
exportColumnAlias [-legacy] -e earPath
```

For example:

```
studio-tools -dt exportToExcel -studioProjPath
D:/be/5.2/examples/standard/WebStudio/CreditCardApplication -dtPath
/Virtual_RF -excelPath C:/temp/exportedDTs -e
D:/be/5.2/examples/standard/WebStudio/CreditCardApplication.ear
```

TIBCO BusinessEvents Studio Tools Options for Exporting Decision Tables to Excel Files

Option	Description
-dt exportToExcel	Within the dt category of operations, specifies the exportToExcel operation for exporting single or all decision tables to Microsoft Excel files.
-h	Optional. Displays help.
-studioProjPath	The file path to the TIBCO BusinessEvents Studio project, from which the decision table is exported.
-dtPath	The project relative path of the decision table to be exported. The path could be for a decision table file or the folder containing decision tables to be exported.
-excelPath	The file path where the exported Excel files are generated.
-useColumnAlias	Optional. Whether to use column alias in the exported Excel files.
-legacy	(Optional) Use this option to export decision table in the .xls format instead of the default .xlsx format.
-e	The file path of the project EAR file.

Importing the Microsoft Excel File to a Decision Table

You can import data from a correctly formatted Microsoft Excel file to create a decision table using the Decision Manager UI.

Some formatted files are provided in the Decision Manager examples shipped with the product.

Note: Row descriptions are imported, but condition and action comments are not imported.

Before you begin

Only import from Excel files is supported, you cannot import from CSV (comma-separated values) files or tab-delimited text files. You can open such files in Excel and save as Excel format binary files. You can use the studio-tools utility to convert CSV files to Excel files. See [Converting a CSV File to a Microsoft Excel File with the Studio Tools Utility](#) for details.

Procedure

1. Do one of the following:
 - In Decision Manager, select **File > Import**.
 - Right-click in the Studio Explorer, and click **Import**.You see the Import dialog.
2. In the **Import** dialog, expand TIBCO BusinessEvents, select Decision Table, and click **Next**.
3. Click **Browse** and select the virtual rule function you want to implement with this decision table.
4. Click **Browse** and select the Microsoft Excel file to import.
5. Type a name for the decision table.
6. Click **Finish**.

Importing the Microsoft Excel File to a Decision Table at the Command Line

You can import a correctly formatted Microsoft Excel file as a decision table using the command line.

Procedure

1. Navigate to *BE_HOME/studio/bin/* and open a command prompt.
2. Execute a command with the following format (all on one line) at a command prompt:

```
studio-tools -dt importExcel [-h] -studioProjPath studioProjectDir [-projWsPath projectWorkspaceDir] -excelPath excelFilePath -dtName decisionTableName -folderPath decisionTableProjectFolderPath -vrfPath vrfProjectFolderPath -wsPath workspaceFolderPath
```

For example:

```
studio-tools -dt importExcel -studioProjPath D:\Workspace\CreditCardApplication -excelPath c:\BankUser.xlsx -dtName BankUser -folderPath \Virtual_RF -vrfPath \Virtual_RF\BankUser_VirtualRuleFunction -wsPath D:\Workspace
```

Reference

importExcel Options

Option	Description
-dt importExcel	Specifies the importExcel operation for importing a Microsoft Excel file to create a decision table.
-h	(Optional) Prints usage help.
-studioProjPath	Specifies the path to the TIBCO BusinessEvents Studio project to which you want to import the file.
-excelPath	Specifies the path to the Microsoft Excel (.xls or .xlsx) file that you

Option	Description
	want to import to TIBCO BusinessEvents Studio.
-dtName	Specifies name of the decision table that you want to create using the Microsoft Excel file.
-folderPath	Specifies the output folder for the created decision table. Specify the output folder path relative to the path of the TIBCO BusinessEvents Studio project path.
-vrfPath	Specifies the path of the virtual rule function that the imported decision table uses. Specify the virtual rule function path relative to the path of the TIBCO BusinessEvents Studio project path and exclude the .vrf extension in the it.
-wsPath	(Optional) Use this option if you want to specify the location of the existing TIBCO BusinessEvents Studio workspace folder to which the decision table is added. The best practice is to leave this option blank so that the project is not stale.

Converting a CSV File to a Microsoft Excel File with the Studio Tools Utility

If you have a CSV (comma-separated values) file of a decision table, you can open and save it in Excel, so that it can be used for import to TIBCO BusinessEvents Decision Manager.

i Note: The Studio Tools utility runs only on Microsoft Windows and Linux platforms as it has dependencies on Eclipse.

Procedure

1. Navigate to `BE_HOME/studio/bin/` and open a command prompt.

2. Execute a command with the following format (all on one line) at a command prompt:

```
studio-tools -dt convertCSV2Excel [-h] -csvPath CSVFilePath -excelPath excelFilePath -s columnSeparator
```

For example:

```
studio-tools -dt convertCSV2Excel -csvPath D:\myfolder\BankUser.csv  
-excelPath D:\myotherfolder\BankUser.xlsx -s ;
```

Result

[Importing the Microsoft Excel File to a Decision Table at the Command Line](#) provides detailed information about the options.

TIBCO BusinessEvents Studio Tools Options for Importing Excel Files into TIBCO BusinessEvents Decision Manager

Option	Description
-dt convertCSV2Excel	Within the dt category of operations, specifies the convertCSV2Excel operation for converting comma separated value (CSV) files to Excel format files.
-h	Optional. Displays help.
-CSVPath	File path to the CSV file to be converted.
-excelPath	File path to the Excel (.xls or .xlsx) file to be created.
-s	Optional. The column separator that is used in the CSV file. Default is ",", (comma).

Microsoft Excel File Format for Decision Tables

The Microsoft Excel file needs to be formatted correctly before importing into the Decision Manager as a decision table.

The following figure shows the sample Microsoft excel file that can be imported as a decision table.

	A	B	C	D	E	F	G	H
1	Version	2						
2								
3	Declarations							
4								
5	Path	Alias	Property					
6	/Concepts/Applic	application	BOTH					
7	/Events/Applicati	applicationreceived	IN					
8								
9	DecisionTable							
10								
11	Id	Condition	Condition	Condition	Action	CustomAction	Description	Priority
12		application.Gender	application.Income	application.Eligible	application.Status			
13	21	Female	< 50000	TRUE	Rejected	sendRejectionNotice()		2
14	31	Female	>= 50000	TRUE	Approved	sendAcceptanceNotice()		3
15	41	Male	>100000	TRUE	Approved	sendAcceptanceNotice()		4
16	51	Male	<= 100000	TRUE	Rejected	sendRejectionNotice()		5
17	61		>1000000	FALSE	NotConsidered	sendAlert()		6
18	71	Female	> 50000	FALSE	Rejected	sendRejectionNotice()		7
19	81	Female	>= 50000	FALSE	Approved	sendAcceptanceNotice()		8
20	91	Male	>100000	TRUE	Approved	sendAcceptanceNotice()		9
21	101	Male	<= 100000	FALSE	Rejected	sendRejectionNotice()		10
22	111	Male	>100000	FALSE	Approved	sendAcceptanceNotice()		1
23	121	Male	<= 100000	FALSE	Approved	sendAcceptanceNotice()		2
24								
25	ExceptionTable							
26								
27	Id	Condition	Condition	Condition	Action	CustomAction	Description	Priority
28		application.Gender	application.Income	application.Age	application.Status			
29	31	UNKNOWN			NotConsidered	sendDetailedRejectionNotice("N		6
30	41		<0		NotConsidered	sendDetailedRejectionNotice("In		7
31	21			<0	NotConsidered	sendDetailedRejectionNotice("M		5
32								

The Excel file format has the following characteristics:

Decision Table Excel File Format

Heading	Notes
Version	Version of this table, as stored in the RMS project.
Declarations Section	
Path	The project path to the entities used as arguments in the VRF
Alias	The entity aliases

Heading	Notes
Property	Not used in this release.
Decision Table Section	
ID	The row (rule) ID
Condition Action Custom Condition Custom Action	One column for each condition and action, as needed. Each regular Condition and Action column heading is followed by the name of the property whose values are used in the cells of the column.
Description	Each rule can have an optional description, seen in the Properties view.
Priority	Rule priority from 1 to 10, seen in the Properties view. Default is 5.
Exception Table Section	
	Same as Decision Table section. Values appear only if an exception table is defined. See Exception Tables .

Decision Tables Validation

TIBCO BusinessEvents Decision Manager validates the decision tables on startup, and every time you save a table. It also provides various tools that you can use to manually check the validity of your decision tables.

On startup, all existing tables are validated for syntax. However, only the delta is validated later.

If there are any access control violations or syntax errors in the tables, they are shown in the **Problems View** tab at the bottom of the application. New errors in syntax are added to these existing errors. Double-click errors to see the problematic view. Take any needed corrective actions and then validate the table again until all errors are resolved.

`NullPointerException`s are silently ignored. They occur when a null `String` is passed to a function that does not check for null, or because you accessed a property of a null contained concept (*parent.child.property* where *child* is null).

If a condition table cell is empty, contains an asterisk, or is disabled, it is skipped and treated as if it evaluates to true.

You can also validate decision tables at the command line. See [Validating a Decision Table at the Command Line](#)

Table Analyzer

Table Analyzer will create a sparse matrix data structure representing an optimized form of the decision table in memory. In the Table Analyzer view you can set example condition values and perform various validation checks on the currently displayed decision table.

i Note:

- Table Analyzer works with decision table view only (and not graph or tree view).
- Table analyzer supports the Not Equal To (!=) operator.

When you open a table, the Table Analyzer view dynamically creates controls for setting values and ranges for each condition in the table.

If the condition is a range, for instance “< 40” or “> 10 && < 100”, then the corresponding control in the Table Analyzer view is a Slider that spans the range from the minimum value to the maximum value. You can then select a range of values within the range. The UP arrow shows the maximum value and the DOWN arrow shows the minimum value in the range.

The Analyze Table Tool

The Analyze Table tool does not use the information in the Table Analyzer view. It analyzes the decision table based on the basic criteria.

Uncovered ranges

Finds gaps in coverage for the selected column's conditions. Suppose a condition column specifies ranges of values for the concept property `client_age`. The table has one row that specifies the range: > 18 && < 25 and another that specifies the range > 30 && < 100. Decision Manager would report that the following values will not trigger an action:

- Values between 25 and 30
- values less than 18
- values greater than 100

Overlapping ranges

Using the client age range example again, suppose you set one condition as > 18 && < 25 and another as > 21 && < 100. Decision Manager would report that these two ranges overlap.

Conflicting Actions

If two rows have identical sets of conditions but different actions, it is detected as a potential issue by the analyzer.

Rules that can be combined

If two rows have different conditions but the same actions, Decision Manager reports that you can combine the conditions. For example, suppose one condition specifies > 50 , and another specifies < 30 , and they both have the same actions. Decision Manager would report that you can combine these conditions as: $> 50 \ || \ < 30$. (greater than fifty or less than thirty.) In this case, you would remove one rule and update the other one accordingly.

How Date Ranges Appear

Dates are treated as numeric ranges. For example the range 20080612 to 20090612 represents the data range for June 12, 2008 to June 12, 2009.

The Show Coverage Tool

When you click the Show Coverage button, Decision Manager highlights all rows in the table that meet all the criteria you specify in the Table Analyzer view.

For example, if you select Male for Gender; true for Eligible; and 50000–60000 for Income, clicking **Show Coverage** displays all rows that have Male, true, and an income between 50000 and 60000.

i Note: If a certain value x is selected for Show Coverage, it will not match any conditions which have value $\neq x$.

For conditions that specify ranges, you can select either of the following behaviors, using an option in Preferences. See [Setting Table Analyzer Preferences](#).

- Decision Manager highlights rows where all of the values specified in the conditions fall within the specified ranges. (This is the default behavior.)

For example, if the slider in the Table Analyzer view is set for a range between 55 and 60, Decision Manager highlights a row whose condition specifies $> 58 \ \&\& \ < 60$. But it does not highlight a row whose condition specifies $> 50 \ \&\& \ < 70$.

- Decision Manager highlights rows where some of the values specified in the conditions fall within the specified ranges. In this case, Decision Manager would display the row from the earlier mentioned example, where the condition specifies $> 50 \ \&\& \ < 70$.

The Generate Test Data Tool

When you click the **Generate Test Data** button, Decision Manager uses the criteria specified in the **Table Analyzer** view to generate one or more lines of test data for the relevant entities to test all possible combinations.

For example, if you select Male and Female for a property Gender, and true and false for a property Eligible, Decision Manager generates four lines of test data that test for all possible combinations:

```
Male, true  
Male, false  
Female, true  
Female, false
```

Auto-Fixing Issues

Table analyzer fixes certain Decision Table's issues automatically.

The table analyzer can take the following actions automatically in case of the respective issue:

- Adds new rules for any uncovered ranges
- Adds uncovered domain entries
- Deletes conflicting action (the later action) for equal set of conditions with different actions

You can auto-fix multiple issues simultaneously by selecting all issues to be fixed.

In problem area double-click issues to be fixed. Click **Ignore...**, if you require no action to be taken.

Setting Table Analyzer Preferences

You can set preferences for table range conditions and to determine if table analyzer appears by default when you display a decision table.

Procedure

1. From the menu select **Window > Preferences**.
2. In the Preferences dialog, select **TIBCO BusinessEvents > Decision Table > Analyzer**.
3. Set your preferences.
See [Decision Table Analyzer Preferences](#) for more details on the preferences.
4. Click **Apply** if you want to set preferences in another dialog before committing all the preferences.
5. Click **OK** to save and close the preferences editor.

Validating a Decision Table Using Test Data

Using test data you can verify if the decision table covers all the data that might be passed through the decision table. This enables you to add more rules to the decision table, if the decision table is not able to cover the test data.

Before you begin

Create the test data for at least one argument from BusinessEvents Studio. See *TIBCO BusinessEvents Developer's Guide* for more information on creation of test data. The test data must be in the project in your repository.

 **Note:** The test data are not managed in the RMS repository.

Procedure

1. In the **Table Analyzer**, click the **Check Test Data Coverage** icon .

 **Note:** You can select the test data for only a single argument at a time.

The Check Test Data Coverage window is displayed with the available test data.

2. Select the test data, for which you want to validate the decision table and click **OK**.

The table analyzer validates the decision table based on the test data and displays the **Test Data Coverage** tab at the bottom pane. The Test Data Coverage pane highlights the rows of test data, which are covered in the decision table.

Validating a Decision Table at the Command Line

You can validate a decision table using the command line.

Note: The Studio Tools utility runs only on Microsoft Windows and Linux platforms as it has dependencies on Eclipse.

Procedure

1. Navigate to *BE_HOME/studio/bin/* and open a command prompt.
2. Execute a command with the following format (all on one line) at a command prompt:

```
studio-tools -dt validateTable [-h] -studioProjPath studioProjectDir [-wsPath workspaceFolderPath] -dtPath decisionTableProjectFolderPath
```

For example:

```
studio-tools -dt validateTable -studioProjPath
D:\Workspace\CreditCardApplication -wsPath D:\Workspace -dtPath
\Virtual_RF\BankUser
```

Reference

validateTable Options

Option	Description
-dt validateTable	Specifies the validateTable operation for generating class files for decision tables.

Option	Description
-h	(Optional) Prints usage help.
-studioProjPath	Specifies the path to the TIBCO BusinessEvents Studio project which contains the decision table that you want to validate.
-dtPath	Specifies path of the decision table that you want to validate. Specify this path relative to the path of the TIBCO BusinessEvents Studio project path and exclude the .dt extension in it.
-wsPath	(Optional) Use this option if you want to specify the location of the existing TIBCO BusinessEvents Studio workspace folder. The best practice is to leave this option blank so that the project is not stale.

Generating and Deploying Decision Table

You can generate EAR files in the Decision Manager user interface, which can be then deployed to be loaded by the engines.

Deployment of EAR files (including hot deployment) is documented in the *TIBCO BusinessEvents Administration* guide of the TIBCO BusinessEvents documentation set.

If RMS is used, you can generate individual decision table class files and deploy them. You can deploy them by placing them in a preconfigured directory. Using JMX, you can also hot deploy class files, and unload class files from a running application. Internally the action is to load class files into the cache, where all engines pick them up. You can also undeploy (unload) decision tables from a running application.

Some configuration is required before you can do these tasks.

i Note: Deployment and unloading of decision table classes requires Cache object management as well as RMS to be used in the TIBCO BusinessEvents application.

Configuring for Deployment and Unloading of Decision Table Classes

Before deployment configure the deployment properties in the CDD file for deployment of classes at application startup, and for hot deployment and unloading of classes.

For class file deployment, including hot deployment, you configure the following items:

- The location where the deployed application looks for the class files for deployment.
- Which engine or engines can and cannot function as *class loaders*.

At application startup, the first class loader that starts will load the generated classes from the preconfigured location to the cache, and other engines pick them up from the cache.

For hot deployment, you use JMX (and RMI for remote invocations) to load the class files into the cache, which causes them to be deployed to all engines.

You can also use JMX to unload class files from a running engine.

Configuring for Loading and Deployment of Decision Table Classes

Generated decision table class files are created in the RMS project directory in the RMS's shared directory.

See [Generated Files Location](#) for default location for generated class files. The location can be configured using properties in the RMS.cdd file (See *TIBCO BusinessEvents WebStudio User Guide* of the TIBCO BusinessEvents documentation set for more details.)

You place these class files in a preconfigured location for deployment. This could be a network location accessible to all engines. A CDD property references this location. If different engines that act as class loaders point to this location differently, add the property at the PU (engine) level and specify it accordingly.

By default all engines act as class loaders. You can prevent specified engines from acting as class loaders using a CDD property.

Procedure

1. In TIBCO BusinessEvents Studio, open the project to which you will deploy the decision table classes.
2. Open the project CDD file in the CDD editor.

3. As needed, add the following properties at the appropriate level of the CDD.

```
be.engine.cluster.externalClasses.path filepath-to-RMS-classes
be.engine.cluster.externalClasses.classLoader true
be.engine.cluster.externalClasses.packageExclusions
```

See the following table for guidelines. The properties are ignored unless there are class files to be deployed, so they are harmless if present and not actually needed. See *TIBCO BusinessEvents Configuration Guide* for details on CDD configuration.

Configuration Properties for Deployment of Decision Table Classes

Property	Notes
be.engine.cluster. externalClasses.path	Specifies the file path to the directory where classes are placed for loading. If engines point to this location differently, for example, because a network drive is mounted differently on the machine where they are running, or if you must use more than one location and copy the class files to all such locations, specify this property as needed at the PU level. Tip: For single-machine installations such as are used for testing or demonstration purposes, you can set the value of this property to the directory specified by the <code>ws.artifact.deploy.location</code> property (See <i>TIBCO BusinessEvents WebStudio User Guide</i> of the TIBCO BusinessEvents documentation set for more details.) When decision tables are generated, they are placed in that directory. The class files would then be automatically available for deployment without manual copying.
be.engine.cluster. externalClasses. classLoader	By default this property is true, so if you want all engines to be able to load classes, do not add the property at all. If you want to prevent certain engines from acting as class loaders, add this property to their PU property sheets and set it to false. Note: Nodes that contain only query agents do not act as

Property	Notes
	class loaders. Possible values: • true • false Default value is true.
<code>be.engine.cluster. externalClasses. packageExclusions</code>	If you will use the core Studio Tools option to generate all class files (see Generating All Project Class Files at the Command Line), you must prevent all class files except decision table class files from being deployed. Add this property to the cluster level property sheet in the project CDD file. Specify the value as a semicolon-delimited list of packages of generated class files to exclude from being deployed.

Configuring for Hot Deployment and Unloading of Decision Table Classes

JMX is a Java standard for managing and monitoring Java applications and services. All agents expose MBeans that can be used from the JConsole application.

An MBean is a managed bean, which is a Java object that can represent a manageable resource.



Note: JMX and JConsole are available in the JDK and not in the JRE provided with TIBCO BusinessEvents. Separate installation of the JDK is required.

Before you can hot deploy or unload decision table classes using JMX, you must do the following in the *BE_HOME/bin/be-engine.tra* files on all machines where the TIBCO BusinessEvents application is (or will be) deployed.

Procedure

1. Ensure that the following entry is present in the `java.extended.properties` property in the engine TRA files of all TIBCO BusinessEvents nodes:

```
-javaagent:%BE_HOME%/lib/cep-base.jar
```

This property is present in the TRA file as default.

2. Add the following properties to the *BE_HOME/bin/be-engine.tra* files on all machines where the TIBCO BusinessEvents application is (or will be) deployed. Configure for your environment as needed.

Enable the `authenticate` property only if you want to enable authentication. The authentication mechanisms available are documented in *TIBCO BusinessEvents Administration* guide of the TIBCO BusinessEvents documentation set.

```
java.property.com.sun.management.jmxremote=true
java.property.com.sun.management.jmxremote.ssl=false
java.property.com.sun.management.jmxremote.port=5558
java.property.com.sun.management.jmxremote.authenticate=false
```

TRA JMX Properties for Hot Deployment and Unloading of Decision Table Classes

Property	Notes
<code>java.property.com.sun.management.jmxremote</code>	Enables remote monitoring. If <code>false</code>, then remote monitoring is not used. Possible values: • <code>true</code> • <code>false</code>
<code>java.property.com.sun.management.jmxremote.ssl</code>	Enables secure monitoring via SSL. If <code>false</code>, then SSL is not used. Possible values: • <code>true</code> • <code>false</code>
<code>java.property.com.sun.management.jmxremote.port</code>	Enables monitoring and management from remote systems on the specified port. Specify an unused port.
<code>java.property.com.sun.management.jmxremote.authenticate</code>	If set to <code>false</code>, JMX does not use passwords or access files. Access is available to all users.

Property	Notes
	Possible values: • true • false

Configuring RMI Properties for Remote Invocation

If you want to invoke MBean operations from a remote client, you must have the RMI registry on the machine that hosts the MBeans. You can host the RMI server on cache servers.

RMI is a Java standard that enables remote communication between Java programs. For an introduction to RMI, see the following:

<http://www.oracle.com/technetwork/java/javase/tech/index-jsp-136424.html>

Procedure

1. In TIBCO BusinessEvents Studio, open the project to which you will deploy the decision table classes.
2. Open the project CDD file in the CDD editor.

3. As needed, configure the following RMI properties at the cluster level of the CDD.

RMI Properties

Property	Description
<code>be.engine.cluster.rmi.enabled</code>	Specifies if RMI functionality is enabled or not. Set to true to enable. The values are: • true • false Default is false
<code>be.engine.cluster.rmi.host</code>	Specifies the host name of the RMI host, that is, the name or IP address of a machine hosting one or more TIBCO BusinessEvents nodes. An RMI server will be started on this machine. Note: Do not use nodes that contain only query agents. The default value is <code>localhost</code>.
<code>be.engine.cluster.rmi.port</code>	Specifies the port to use when starting the RMI server. If a machine has two TIBCO BusinessEvents processing units (PUs), and you want to start two RMI servers, one for each PU, specify a different port for each PU. Default is 9999.

Generating Deployable Files (EAR and Class Files)

Deployable files are generated using the TIBCO BusinessEvents Studio project files, located in the RMS project's folder.

The deployable files are of two types: EAR files and class files.

EAR files are generated in the Decision Manager UI. It is also possible to use standard TIBCO BusinessEvents tools for generating EAR files, as documented in *TIBCO BusinessEvents Developer's Guide* of the TIBCO BusinessEvents documentation set.

You can generate class files using the Decision Manager UI, or at the command line. You can generate an entire project's class files, or one decision table's class file. If you generate an entire project's class files, a property in the CDD file enables you to exclude unwanted packages.

i Note: You can generate one decision table's class file using TIBCO BusinessEvents WebStudio only. See the *TIBCO BusinessEvents WebStudio Users Guide* of the TIBCO BusinessEvents documentation set for more details.

Generating the Project EAR or All Project Class Files

You can generate deployable EAR files only if you have permission to check out all the required project resources (ACLs may limit what you can check out).

Generated files are saved to the preconfigured location (see [Generated Files Location](#)). Any existing files are overwritten.

Procedure

1. From the RMS menu, select **Generate Deployable**.
You see the **Generate Deployable** dialog.
2. Do one of the following:
 - To generate class files, select the **Generate Classes Only** check box.
 - To generate EAR files, clear the **Generate Classes Only** check box.
3. From the drop-down list, select the project for which you want to generate the deployable files.
4. (For EAR files only) If you want to generate the debug information, select the **Generate Debug Info** check box.
5. (For EAR files only) If you want to include all service-level global variables, check the check box.
6. Click **OK**.

Generating One Decision Table's Class File at the Command Line

Without WebStudio, you can generate the decision table's class file only using command line.

Procedure

1. Navigate to *BE_HOME/studio/bin/* and open a command prompt.

2. Execute a command with the following format (all on one line) at a command prompt:

```
studio-tools -dt generateDTClass [-h] -d DTProjectPath -p studioProjectDir -o
outputPath -e EARPath [-x {true | false}] [-pl projectLibraryFilePath] [-cp
extendedClasspath]
```

For Example:

```
studio-tools -dt generateDTClass -d rules/myVRF/myimpl -p
D:\Workspace\FraudDetection -o D:\Workspace\FraudDetection -e
c:\temp -x true -cp c:\tibco\be
\5.2\lib\myjar.jar
```

The following table provides detailed information about the options.

TIBCO BusinessEvents Studio Tools for Generating One Table's Class Files

Option	Description
-dt generateDTClass	Within the dt category of operations, specifies the generateDTClass operation for building EAR files.
-h	Optional. Displays help.
-d	Project path to the decision table.
-p	Absolute path to the TIBCO BusinessEvents Studio project directory.
-o	Specifies the output directory for generated classes. If you do not specify a directory, files are placed in a user temporary directory. For example, on Windows files might go in a directory like the following: C:\Documents and Settings\User\Local Settings\Temp\BE_1322046141896
-e	Specifies path to the EAR files of the project which contains the

Option	Description
	decision table.
-x	Optional. If true, overwrites any existing class file with the same name.
-pl	Optional. Specifies list of project library file paths to be used, separated by a path separator.
-cp	Optional. Extended classpath. Use as needed. Provide separate JAR file paths for each JAR file required for project compilation. For example, additional classpath information is needed if the decision table uses custom functions or third-party JAR files. Separate entries by the appropriate path separator. For example if the separator is semicolon (;) you might add the following: C:\customjars\custom.jar;C:\customjars\custom2.jar

You see a success message if the files were generated successfully.

Generating All Project Class Files at the Command Line

You can generate all class files in a project at the command line. Although this is a core component, the class files are generally used within the context of TIBCO BusinessEvents Decision Manager, where decision table class files can be separately deployed.

i Note: This is a TIBCO BusinessEvents tool and it does not by default save classes to the RMS project directory. Instead you specify a location in the command.

You must place the class files in the location where a class loader picks up files for loading, and you must exclude all unwanted files from being loaded. See [Generated Files Location](#) for details.

Procedure

1. Navigate to *BE_HOME*/studio/bin/ and open a command prompt.

2. Execute a command with the following format (all on one line) at a command prompt:

```
studio-tools -core generateClass [-h] -p studioProjectDir [-n
studioProjectName] -o outputPath [-x {true | false}] [-lc] [-pl
projectLibraryFilePath] [-cp extendedClasspath]
```

For example:

```
studio-tools -core generateClass -p D:\Workspace\FraudDetection -o
c:\temp -x true -cp c:\tibco\be\5.2\lib\myjar.jar
```

The following table provides detailed information about the options.

TIBCO BusinessEvents Studio Tools for Generating Class Files

Option	Description
-core generateClass	Within the core category of operations, specifies the generateClass operation used to generate a project's class files.
-h	Optional. Displays help.
-p	Absolute path to the TIBCO BusinessEvents Studio project directory.
-n	Optional. Specifies the name of the TIBCO BusinessEvents Studio project whose class files are to be generated. If not specified, the final (leaf) directory name in the path specified for the -p option is used as the project name.
-o	Specifies the output directory for generated classes. If you do not specify a directory, files are placed in a user temporary directory. For example, on Windows files might go in a directory like the following: C:\Documents and Settings\User\Local Settings\Temp\BE_1322046141896

Option	Description
-x	Optional. If <code>true</code> , overwrites any existing class file with the same name.
-lc	Optional. Specifies that the file-based legacy compiler must be used to build the EAR file. By default, the EAR files are built-in memory.
-pl	Optional. Specifies list of project library file paths to be used, separated by a path separator.
-cp	Optional. Extended classpath. Use as needed. Provide separate JAR file paths for each JAR file required for project compilation. For example, additional classpath information is needed if the decision table uses custom functions or third-party JAR files. Separate entries by the appropriate path separator. For example if the separator is semicolon (;) you might add the following: <pre>C:\customjars\custom.jar;C:\customjars\custom2.jar</pre>

You see a success message if the files were generated successfully.

Deploying and Unloading Decision Table Class Files

Advisory events for success and failure are asserted as needed during agent startup, and JMX-based hot deployment.

i Note: Deployment and unloading of decision table classes requires Cache object management to be used in the TIBCO BusinessEvents application.

Hot Deploying External Classes Using JMX

After you have configured the system and generated decision table class files for deployment, you can hot deploy the classes to running TIBCO BusinessEvents engines using JMX.

You can also unload a deployed decision table from a running engine. The system pauses briefly until the procedure is complete.

You might want to use these techniques, for example if you want to replace one VRF implementation (decision table) with a different one.

Procedure

1. Open a command window in the bin directory of your JDK installation and type **jconsole**.

You see the **New Connection** dialog.

2. Select the connection corresponding to the TIBCO BusinessEvents node that you want to view and click **Connect**. Each node runs in single JVM.

You see the multi-tabbed **Connection** dialog.

3. Select the **MBeans** tab and expand the tree nodes on the left to **com.tibco.be > > Cluster > Operations**.

4. Do the following as appropriate:

- Click **loadAndDeploy**. All decision table class files in the preconfigured location are deployed to all nodes in the cluster.
- Specify the project path to the virtual rule function and the decision table name and click **loadAndDeploy**. The corresponding class is loaded into the cache and deployed to all engines.
- Specify the project path to the virtual rule function and the decision table name and click **unloadClass**. The corresponding class is unloaded.

Advisory Events for Deployment Success and Failure

Advisory events are asserted for every inference agent for every class successfully loaded or failed to load at runtime.

Advisory events are asserted for every class that failed to load, in cases where there is a failure on an agent while loading a class. The advisory events are asserted as needed during agent startup, and JMX-based hot deployment.

Advisory Events for Deployment

Deployment Result	Category	Type	Message
Success	Deployment	deployment.externalclasses.success	External class <code>className</code> deployment successful
Failure	Deployment	deployment.externalclasses.failure	External class <code>className</code> deployment on member failed

See the chapter on advisory events in *TIBCO BusinessEvents Developer's Guide* of the TIBCO BusinessEvents documentation set for details about using advisory events in rules.

Generated Files Location

Location of generated files can be configured using deploy location properties (`ws.artifact.deploy.location`) in the `RMS.cdd` file.

See *TIBCO BusinessEvents WebStudio User Guide* of the TIBCO BusinessEvents documentation set for more details on deploy location properties. Following are the location of generated files as per the preconfigured location in the shipped product.

- EAR files are saved in the RMS project directory in the RMS's shared directory. For example, EAR file for CreditCardApplication project (`CreditCardApplication.ear`) is stored at

`BE_HOME\rms\shared\CreditCardApplication`

- Project's class files are saved in the `codegen` subdirectory under the RMS project directory in the RMS's shared directory. For example, class files for CreditCardApplication project (`CreditCardApplication.ear`) are stored at `BE_HOME\rms\shared\CreditCardApplicatio\codegen`
- Individual decision table's class file is saved in the RMS project directory in the RMS's shared directory.

TIBCO Documentation and Support Services

For information about this product, you can read the documentation, contact TIBCO Support, and join TIBCO Community.

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.

Product-Specific Documentation

The following documentation for this product is available on the [TIBCO BusinessEvents® Enterprise Edition Product Documentation](#) page:

- *TIBCO BusinessEvents® Release Notes*
- *TIBCO BusinessEvents® Installation*
- *TIBCO BusinessEvents® Getting Started*
- *TIBCO BusinessEvents® Architect Guide*
- *TIBCO BusinessEvents® Administration*
- *TIBCO BusinessEvents® Developer Guide*
- *TIBCO BusinessEvents® Configuration Guide*
- *TIBCO BusinessEvents® Migration Guide*
- *TIBCO BusinessEvents® Data Modeling Developer Guide*
- *TIBCO BusinessEvents® Decision Manager User Guide*
- *TIBCO BusinessEvents® WebStudio User Guide*
- *TIBCO BusinessEvents® Event Stream Processing Pattern Matcher Developer Guide*
- *TIBCO BusinessEvents® Event Stream Processing Query Developer Guide*

- *TIBCO BusinessEvents® Security Guide*
- Online References:
 - *TIBCO BusinessEvents® Java API Reference*
 - *TIBCO BusinessEvents® Functions Reference*

To directly access documentation for this product, double-click the file at the following location:

`TIBCO_HOME/release_notes/TIB_businessevents-enterprise_6.3.0_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`.

Other TIBCO Product Documentation

When working with TIBCO BusinessEvents Enterprise Edition, you may find it useful to read the documentation of the following TIBCO products:

- TIBCO ActiveSpaces®: It is used as the cluster, cache, or store provider for the TIBCO BusinessEvents Enterprise Edition project.
- TIBCO FTL®: It is used as the cluster provider for the TIBCO BusinessEvents Enterprise Edition project.

How to Access Related Third-Party Documentation

When working with TIBCO BusinessEvents® Enterprise Edition, you may find it useful to read the documentation of the following third-party products:

- Apache Ignite
- Apache Kafka
- Confluent Kafka Schema Registry
- TIBCO Messaging - Schema Repository for Apache Kafka
- Apache Pulsar
- GridGain
- Apache Cassandra

- Grafana
- InfluxDB
- OpenTelemetry

How to Contact TIBCO Support

Get an overview of [TIBCO Support](#). You can contact TIBCO Support in the following ways:

- For accessing the Support Knowledge Base and getting personalized content about products you are interested in, visit the [TIBCO Support](#) website.
- 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 [TIBCO Support](#) website. 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 [TIBCO Community](#).

Legal and Third-Party Notices

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE “LICENSE” FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of Cloud Software Group, Inc.

TIBCO, the TIBCO logo, the TIBCO O logo, TIBCO BusinessEvents, ActiveMatrix, ActiveMatrix BusinessWorks, ActiveSpaces, TIBCO Administrator, TIBCO Designer, Enterprise Message Service, TIBCO FTL, Hawk, and TIBCO Runtime Agent are either registered trademarks or trademarks of Cloud Software Group, Inc. in the United States and/or other countries.

Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

This document includes fonts that are licensed under the SIL Open Font License, Version 1.1, which is available at: <https://scripts.sil.org/OFL>

Copyright (c) Paul D. Hunt, with Reserved Font Name Source Sans Pro and Source Code Pro.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

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. See the readme 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. CLOUD SOFTWARE GROUP, 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 Cloud Software Group, Inc. may be covered by registered patents. Please refer to TIBCO's Virtual Patent Marking document (<https://www.tibco.com/patents>) for details.

Copyright © 2004-2023. Cloud Software Group, Inc. All Rights Reserved.