

TIBCO BusinessEvents™

Getting Started

*Software Release 4.0
May 2010*

The Power to Predict™

 **TIBCO®**
The Power of Now®

Important Information

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 LICENSE.PDF) 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 contains confidential information that 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 TIBCO Software Inc.

TIB, TIBCO, TIBCO Software, TIBCO Adapter, Predictive Business, Information Bus, The Power of Now, The Power to Predict, TIBCO BusinessEvents, TIBCO ActiveSpaces, TIBCO ActiveMatrix BusinessWorks, TIBCO Rendezvous, TIBCO Enterprise Message Service, TIBCO PortalBuilder, TIBCO Administrator, TIBCO Runtime Agent, TIBCO General Interface, and TIBCO Hawk are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

EJB, Java EE, J2EE, JMS and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

Excerpts from Oracle Coherence documentation are included with permission from Oracle and/or its affiliates. Copyright © 2000, 2006 Oracle and/or its affiliates. All rights reserved.

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.TXT FILE FOR THE AVAILABILITY OF THIS SOFTWARE VERSION ON A SPECIFIC OPERATING SYSTEM PLATFORM.

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

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

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

This product is covered by U.S. Patent No. 7,472,101.

Copyright © 2004-2010 TIBCO Software Inc. ALL RIGHTS RESERVED.

TIBCO Software Inc. Confidential Information

Contents

Preface	v
Related Documentation	vi
TIBCO BusinessEvents Documentation	vi
TIBCO BusinessEvents Event Stream Processing	vii
TIBCO BusinessEvents Decision Manager	vii
TIBCO BusinessEvents Data Modeling	vii
TIBCO BusinessEvents Views	viii
Other TIBCO Product Documentation	viii
Typographical Conventions	ix
How to Contact TIBCO Support	xii
 Chapter 1 Introduction	 1
Overview	2
A Configured Tutorial Project is Provided	2
Additional Example Projects Provide More Learning	2
Skills Required	2
The Fraud Detection Scenario	3
 Chapter 2 Project Design and Deployment Tutorial	 5
General Runtime Flow	6
Create the FraudDetection Project	7
Create an HTTP Channel and Destination	10
Define the AccountOperation, CreateAccount, Debit, and Reply Events	13
Define the Account Concept	17
Add the FraudCriteria Scorecard	20
Add the InitializeScorecard Rule Function	22
Add the PreProcessor Rule Function	24
Add BadCreateAccount and CreateAccount Rules	26
Add ApplyDebit, BadApplyDebit, and CheckNegativeBalance Rules	30
Add the FraudDetection Rule and Unsuspend Account Rule	34
Analyze and Validate the Project	37
Add a Cluster Deployment Descriptor and Build the EAR File	40
Start the Engine and Run the Application	43

Glossary 45

Index 51

Preface

TIBCO BusinessEvents™ allows you to abstract and correlate meaningful business information from the events and data flowing through your information systems, and take appropriate actions using business rules. By detecting patterns within the real-time flow of events, BusinessEvents™ can help you to detect and understand unusual activities as well as recognize trends, problems, and opportunities. BusinessEvents publishes this business-critical information in real time to your critical enterprise systems or dashboards. With BusinessEvents you can predict the needs of your customers, make faster decisions, and take faster action.

BusinessEvents
The Power to Predict™

Topics

- [Related Documentation, page vi](#)
- [Typographical Conventions, page ix](#)
- [How to Contact TIBCO Support, page xii](#)

Related Documentation

This section lists documentation resources you may find useful.

TIBCO BusinessEvents Documentation

- *TIBCO BusinessEvents Installation*: Read this manual for instructions on site preparation and installation.
- *TIBCO BusinessEvents Getting Started*: After the product is installed, use this manual to learn the basics of BusinessEvents. This guide provides step-by-step instructions to implement an example project and also explains the main ideas so you gain understanding as well as practical knowledge.
- *TIBCO BusinessEvents Architect's Guide*: If you are architecting an application using TIBCO BusinessEvents, read this guide for overview and detailed technical information to guide your work.
- *TIBCO BusinessEvents Developer's Guide*: After the architect has designed the system, use this manual to implement the design in BusinessEvents Studio.
- *TIBCO BusinessEvents Administration*: This book explains how to configure, deploy, monitor, and manage a BusinessEvents application and the data it generates.
- Online References:
 - *TIBCO BusinessEvents Cache Configuration Guide*: This online reference is available from the HTML documentation interface. It provides configuration details for cache-based object management. Cache-based object management is explained in *TIBCO BusinessEvents Administration*.
 - *TIBCO BusinessEvents Java API Reference*: This online reference is available from the HTML documentation interface. It provides the Javadoc-based documentation for the BusinessEvents API.
 - *TIBCO BusinessEvents Functions Reference*: This online reference is available from the HTML documentation interface. It provides a listing of all functions provided with BusinessEvents, showing the same details as the tooltips available in the BusinessEvents Studio rule editor interface.
- *TIBCO BusinessEvents Release Notes*: Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

TIBCO BusinessEvents Event Stream Processing

This BusinessEvents add-on is available separately, and includes the BusinessEvents Query Language features and the Pattern Matching Framework.

- *TIBCO BusinessEvents Event Stream Processing Installation*: Read this brief manual for installation instructions. A compatible version of TIBCO BusinessEvents must be installed first.
- *TIBCO BusinessEvents Query Developer's Guide*: This manual explains how to use the object query language to query various aspects of the running system.
- *TIBCO BusinessEvents Event Stream Processing Pattern Matcher Developer's Guide*: This manual explains how to use the pattern matcher language and engine to correlate event patterns in a running system.
- *TIBCO BusinessEvents Event Stream Processing Release Notes*: Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

TIBCO BusinessEvents Decision Manager

This BusinessEvents add-on is available separately. It incorporates a decision modeling business user interface, and associated runtime.

- *TIBCO BusinessEvents Decision Manager Installation*: Read this brief manual for installation instructions. A compatible version of TIBCO BusinessEvents must be installed first.
- *TIBCO BusinessEvents Decision Manager User's Guide*: This manual explains how business users can use decision tables and other decision artifacts to create business rules. It also covers configuration and administration of Rules Management Server, which is used for authentication, authorization, and approval processes.
- *TIBCO BusinessEvents Decision Manager Release Notes*: Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

TIBCO BusinessEvents Data Modeling

This BusinessEvents add-on is available separately. It contains state models and database concept features.

- *TIBCO BusinessEvents Data Modeling Installation*: Read this brief manual for installation instructions. A compatible version of TIBCO BusinessEvents must be installed first.

- *TIBCO BusinessEvents Data Modeling Developer's Guide*: This manual explains data modeling add-in features for BusinessEvents. The database concepts feature enables you to model BusinessEvents concepts on Database tables. The state modeler feature enables you to create state machines.
- *TIBCO BusinessEvents Data Modeling Release Notes*: Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

TIBCO BusinessEvents Views

This BusinessEvents add-on is available separately. It includes graphical dashboard components for run-time event monitoring.

- *TIBCO BusinessEvents Views Installation*: Read this manual for instructions on site preparation and installation.
- *TIBCO BusinessEvents Views Developer's Guide*: This book explains how to use BusinessEvents Views to create meaningful metrics that are presented to business users in real-time for proactive decision making.
- *TIBCO BusinessEvents Views User's Guide*: This book explains how to monitor metrics in BusinessEvents Views and how to represent the business processes graphically.
- *TIBCO BusinessEvents Views Release Notes*: Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

Other TIBCO Product Documentation

You may find it useful to refer to the documentation for the following TIBCO products:

- TIBCO ActiveSpaces[®]
- TIBCO Hawk[®]
- TIBCO Rendezvous[®]
- TIBCO Enterprise Message Service[™]
- TIBCO ActiveMatrix BusinessWorks[™]

Typographical Conventions

The following typographical conventions are used in this manual.

Table 1 General Typographical Conventions

Convention	Use
<i>TIBCO_HOME</i> <i>ENV_HOME</i> <i>BE_HOME</i>	<p>Many TIBCO products must be installed within the same home directory. This directory is referenced in documentation as <i>TIBCO_HOME</i>. The value of <i>TIBCO_HOME</i> depends on the operating system. For example, on Windows systems, the default value is C:\tibco.</p> <p>Other TIBCO products are installed into an installation environment. Incompatible products and multiple instances of the same product are installed into different installation environments. The directory into which such products are installed is referenced in documentation as <i>ENV_HOME</i>. The value of <i>ENV_HOME</i> depends on the operating system. For example, on Windows systems the default value is C:\tibco.</p> <p>TIBCO BusinessEvents installs into a directory within <i>ENV_HOME</i>. This directory is referenced in documentation as <i>BE_HOME</i>. The value of <i>BE_HOME</i> depends on the operating system. For example on Windows systems, the default value is C:\tibco\TIBCO BusinessEvents\2.1.</p>
code font	<p>Code font identifies commands, code examples, filenames, pathnames, and output displayed in a command window. For example:</p> <p>Use MyCommand to start the foo process.</p>
bold code font	<p>Bold code font is used in the following ways:</p> <ul style="list-style-type: none"> • In procedures, to indicate what a user types. For example: Type admin. • In large code samples, to indicate the parts of the sample that are of particular interest. • In command syntax, to indicate the default parameter for a command. For example, if no parameter is specified, MyCommand is enabled: MyCommand [enable disable]

Table 1 General Typographical Conventions (Cont'd)




Convention	Use
<i>italic font</i>	<p>Italic font is used in the following ways:</p> <ul style="list-style-type: none">• To indicate a document title. For example: See <i>TIBCO BusinessWorks Concepts</i>.• To introduce new terms For example: A portal page may contain several <i>portlets</i>. Portlets are mini-applications that run in a portal.• To indicate a variable in a command or code syntax that you must replace. For example: <code>MyCommand <i>pathname</i></code>
Key combinations	<p>Key name separated by a plus sign indicate keys pressed simultaneously. For example: Ctrl+C.</p> <p>Key names separated by a comma and space indicate keys pressed one after the other. For example: Esc, Ctrl+Q.</p>
	The note icon indicates information that is of special interest or importance, for example, an additional action required only in certain circumstances.
	The tip icon indicates an idea that could be useful, for example, a way to apply the information provided in the current section to achieve a specific result.
	The warning icon indicates the potential for a damaging situation, for example, data loss or corruption if certain steps are taken or not taken.

Table 2 Syntax Typographical Conventions

Convention	Use
[]	<p>An optional item in a command or code syntax.</p> <p>For example:</p> <pre>MyCommand [optional_parameter] required_parameter</pre>
	<p>A logical 'OR' that separates multiple items of which only one may be chosen.</p> <p>For example, you can select only one of the following parameters:</p> <pre>MyCommand param1 param2 param3</pre>

Table 2 *Syntax Typographical Conventions*

Convention	Use
{ }	<p>A logical group of items in a command. Other syntax notations may appear within each logical group.</p> <p>For example, the following command requires two parameters, which can be either the pair param1 and param2, or the pair param3 and param4.</p> <pre>MyCommand {param1 param2} {param3 param4}</pre> <p>In the next example, the command requires two parameters. The first parameter can be either param1 or param2 and the second can be either param3 or param4:</p> <pre>MyCommand {param1 param2} {param3 param4}</pre> <p>In the next example, the command can accept either two or three parameters. The first parameter must be param1. You can optionally include param2 as the second parameter. And the last parameter is either param3 or param4.</p> <pre>MyCommand param1 [param2] {param3 param4}</pre>

How to Contact TIBCO Support

For comments or problems with this manual or the software it addresses, please contact TIBCO Support as follows.

- For an overview of TIBCO Support, and information about getting started with TIBCO Support, visit this site:

<http://www.tibco.com/services/support>

- If you already have a valid maintenance or support contract, visit this site:

<https://support.tibco.com>

Entry to this site requires a user name and password. If you do not have a user name, you can request one.

Chapter 1 **Introduction**

This brief chapter provides an overview of the tutorial provided in this guide.

Topics

- [Overview, page 2](#)
- [The Fraud Detection Scenario, page 3](#)

Overview

This guide contains a tutorial based on one simplified business scenario. The tasks in the tutorial provide step-by-step instructions and also explain the main ideas so you gain understanding as well as practical knowledge. References to related information are provided so you can jump to the main documentation on any topic to learn more.

The tutorial shows you how to configure a BusinessEvents project, run it at the command line, and test its behavior. Using a simple scenario, this tutorial focuses on ontology and inferencing features.

A Configured Tutorial Project is Provided

The `FraudDetection` example is located in the `BE_HOME/examples/standard` directory.

Additional Example Projects Provide More Learning

After you have completed the tutorials, you can explore the examples in the `BE_HOME/Examples` directory. These examples demonstrate specific techniques that you can apply in your work.

Skills Required

The tutorial is written for users with little or no familiarity with TIBCO products. Readers should have some familiarity with Java programming and the Eclipse platform.

Eclipse Platform

If you are not familiar with Eclipse platform, it is recommended that you first read the topics available from the Overview link on the Welcome page, especially Workbench basics, which explains concepts such as perspectives and views. Then complete the "Create a Hello World Application" available from the Tutorials link on the Welcome page, and any other tutorials of interest.

The Fraud Detection Scenario

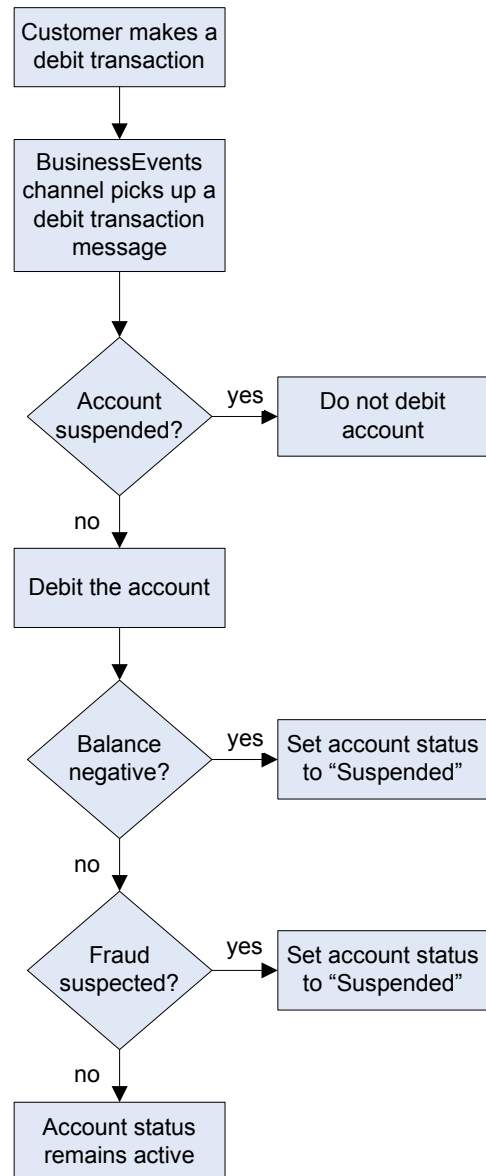
The tutorial is built on a simplified fraud detection scenario and decision making flow, illustrated by the following flow chart.

To establish whether fraud is suspected, the runtime engine correlates the frequency of and amount of debits in a rolling time window, and flags accounts that satisfy both of the following criteria:

- The account incurs more than three debit transactions in a two minute period.
- The sum of the debits that occurred in the two minute period totals more than 80% of the average monthly balance of the account.

For the purpose of the tutorial, messages arrive from an HTTP server, provided by BusinessEvents. The event correlation is performed using simple BusinessEvents rules. Suspicious accounts are simply set to "Suspended." Actions are printed to the monitor so you can see the project in action.

An additional event and rule not shown allow you to unsuspend an account.



Chapter 2

Project Design and Deployment Tutorial

This tutorial takes you through all the steps of configuring, building, deploying, and testing a BusinessEvents project. The emphasis is on basic project design, with deploytime activities limited to the basic actions required to test a design.

If you don't want to complete the steps, you can refer to the fully configured example project, provided here:

`BE_HOME/examples/FraudDetection`

Topics

- [General Runtime Flow, page 6](#)
- [Create the FraudDetection Project, page 7](#)
- [Create an HTTP Channel and Destination, page 10](#)
- [Define the AccountOperation, CreateAccount, Debit, and Reply Events, page 13](#)
- [Define the Account Concept, page 17](#)
- [Add the FraudCriteria Scorecard, page 20](#)
- [Add the InitializeScorecard Rule Function, page 22](#)
- [Add the PreProcessor Rule Function, page 24](#)
- [Add BadCreateAccount and CreateAccount Rules, page 26](#)
- [Add ApplyDebit, BadApplyDebit, and CheckNegativeBalance Rules, page 30](#)
- [Add the FraudDetection Rule and Unsuspend Account Rule, page 34](#)
- [Analyze and Validate the Project, page 37](#)
- [Add a Cluster Deployment Descriptor and Build the EAR File, page 40](#)
- [Start the Engine and Run the Application, page 43](#)

General Runtime Flow

The section [The Fraud Detection Scenario on page 3](#) explains the general tutorial scenario, the fraud detection criteria, and how a customer account can become Suspended. This section explains in more technical terms what happens at runtime given an example debit that triggers the fraud detection rules. (You will learn more details about the terms shown in *italics* below, as you complete the tutorial steps):

1. A message arriving through a BusinessEvents *channel* is transformed into an *event*. (At design time you create an event type for this purpose, with the appropriate properties.) The event instance is then *asserted into the Rete network*, an in-memory network of objects based on the Rete algorithm which enables fast matching of facts with rule dependencies.
2. The presence of this new event in the Rete network causes the inference engine to check for rules that are designed to be triggered when this event is asserted.
3. A rule that has is triggered by this event executes. It may make changes to concept instances, create an event and send it to a channel (and out of the BusinessEvents application to some destination), and so on. The rule then consumes the event unless there is a reason not to do so.



Event lifespan It is important to consume events when they are no longer needed so that they don't trigger rules to fire erroneously. On the other hand, it is also important to use a long enough time-to-live (TTL) setting for an event, so that it exists long enough to perform all work needed, for example, to trigger rules that correlate multiple events and take appropriate actions.

Create the FraudDetection Project

In this task, you start BusinessEvents Studio and create an empty project.

Learning Points

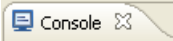
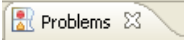
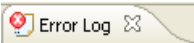

What is BusinessEvents Studio? BusinessEvents Studio is the Eclipse-based user interface for BusinessEvents. It enables you to build, test, and debug projects. Use of this industry-standard development framework shortens your learning curve and enables you to take advantage of common tools and facilities.

What are the BusinessEvents Perspectives? BusinessEvents has these Eclipse perspectives:

- BusinessEvents Studio Debug
- BusinessEvents Studio Development
- BusinessEvents Studio Diagram

BusinessEvents switches perspectives transparently depending on the editor you are working with.

What are the bottom tabs? In the lower part of the user interface are various tabs used as needed:

Tab	Description
	Console The Console view displays and logs the errors resulting from improper execution of the BusinessEvents engine.
	Problems The Problems view reports all validation errors in the project, including language validation errors, access control errors, and so on. It also displays messages relating to project analyzer. Double-click a problem entry to open the associated editor. See Analyze and Validate the Project on page 37 for more details.
	Error Log The Error Log view captures all the warnings and errors logged by Eclipse plug-ins. The log file itself has the extension <code>.log</code> file and it is stored in the <code>.metadata</code> subdirectory of the workspace.
	Properties The Properties view shows metadata about a resource. To view the metadata, click the resource in Studio Explorer. Not all resources use this tab.

How should I organize project folders? When you create a new project a set of folders is created. You can use a different folder structure to organize your project components in any way you like. Example projects are kept simple and use provided folder names such as Concepts, Events, Rules. More complex projects might use a different folder hierarchy with names that relate to the purpose or contents of the folders.



Incorrect display? If the Eclipse user interface is not rendering correctly, try adjusting your computer display options. For example, on Windows XP, the following option causes rendering problems: Control Panel > System Properties > Advanced > Performance Options > Visual Effects > Adjust for best performance. In this example, choosing the Adjust for best performance option results in correct rendering.

More Information

Chapter 2, Project Tasks in *TIBCO BusinessEvents Developer's Guide* discusses actions you can take at the project level such as importing projects from an earlier release, validating projects, using project libraries, and so on.

Task A Create the Fraud Detection Project

1. Click **Start > All Programs > TIBCO > TIBCO BusinessEvents 4.0> BusinessEvents Studio**. (Your TIBCO Home may differ.)
2. The first time you run BusinessEvents Studio, a Welcome screen displays. You can access this guide from the Welcome page. Click the X next to Welcome to dismiss the screen.
3. Right click in the BusinessEvents Studio Explorer view (the panel on the left, where the project folders will display), and select **New > Project**. You see the New Project wizard.
4. From the list, select **TIBCO BusinessEvents > Studio Project**.
Then click **Next**.
5. In the New Studio Project dialog, enter the project name **FraudDetection**, and click **Finish**. (If you want to use a non-default location, uncheck the Use default location checkbox and select the location.)

In the Studio Explorer viewer, the root folder of the project is called FraudDetection. It has a set of project subfolders.

In addition you see links for various diagrams:

`FraudDetection.conceptview`, `FraudDetection.eventview` and `FraudDetection.projectview`. These are respectively concept model, event model, and project (or element) diagrams. diagrams are created dynamically and not saved. More diagrams are available for other purposes, as you will see. They are UML compliant, with some exceptions.

6. Save the project.
 - To save all changes to all resources in a project (since last save), click **File > Save All** or click **Ctrl+Shift+S**.
 - To save changes in just the currently viewed resource, click **File > Save** or click **Ctrl+S**, or click the **Save** button.

Summary and Next Steps

You have created a new empty project in the BusinessEvents Studio Development perspective.

Next you will begin to define your BusinessEvents project by building a channel for information to enter the deployed application, and a destination for the application to listen to.

The order in which you build up the project is not fixed. For example, instead you might define the project ontology first.

Create an HTTP Channel and Destination

In this task you configure an HTML channel with one destination. The `AllOps` destination listens for messages that come from HTTP forms embedded in the project's `readme.html` file.

Example projects use the HTTP channel because it does not require use of any external software. If you have TIBCO ActiveMatrix BusinessWorks, TIBCO Enterprise Message Service, TIBCO Rendezvous, or other source for messages you can experiment with adding different types of channels.

Learning Points

What are channels and destinations? Messages enter and leave the system through channels. You create destinations within a channel to define the message sources and sinks. BusinessEvents events are created using data in incoming messages. Outgoing messages are created using data from events. Later in the tutorial, you will set up the relationship between these destinations and the event types that they listen to by default.

Note that in this tutorial outbound messages are simply sent to the console, so there are no outbound destinations.

How are channels and destinations created? You create channels and destinations at design-time, as explained below. When you are planning BusinessEvents projects, you would consider the incoming and outgoing messages for your project, and then define the channels, destinations, and the corresponding event types — outbound events are transformed into appropriate messages, and inbound messages are transformed into events of a specified type.

Why Use Shared Resources? Shared resources are generally used to configure communication with some external system such as a database server or JMS server. You can configure the connection once and use it in multiple places. If some configuration has to be changed, you just have to change it in one place.

More Information

Chapter 4, Channels and Destinations, and Chapter 5, JMS Channels in *TIBCO BusinessEvents Developer's Guide*.

Task B Create the HTTP Connection

1. Select the `SharedResources` folder and press **Ctrl+N**. You see the Select a Wizard dialog. (You could also get here using `File > New > Other`.)
2. Select **TIBCO Shared Resources > HTTP Connection** and click **Next**.

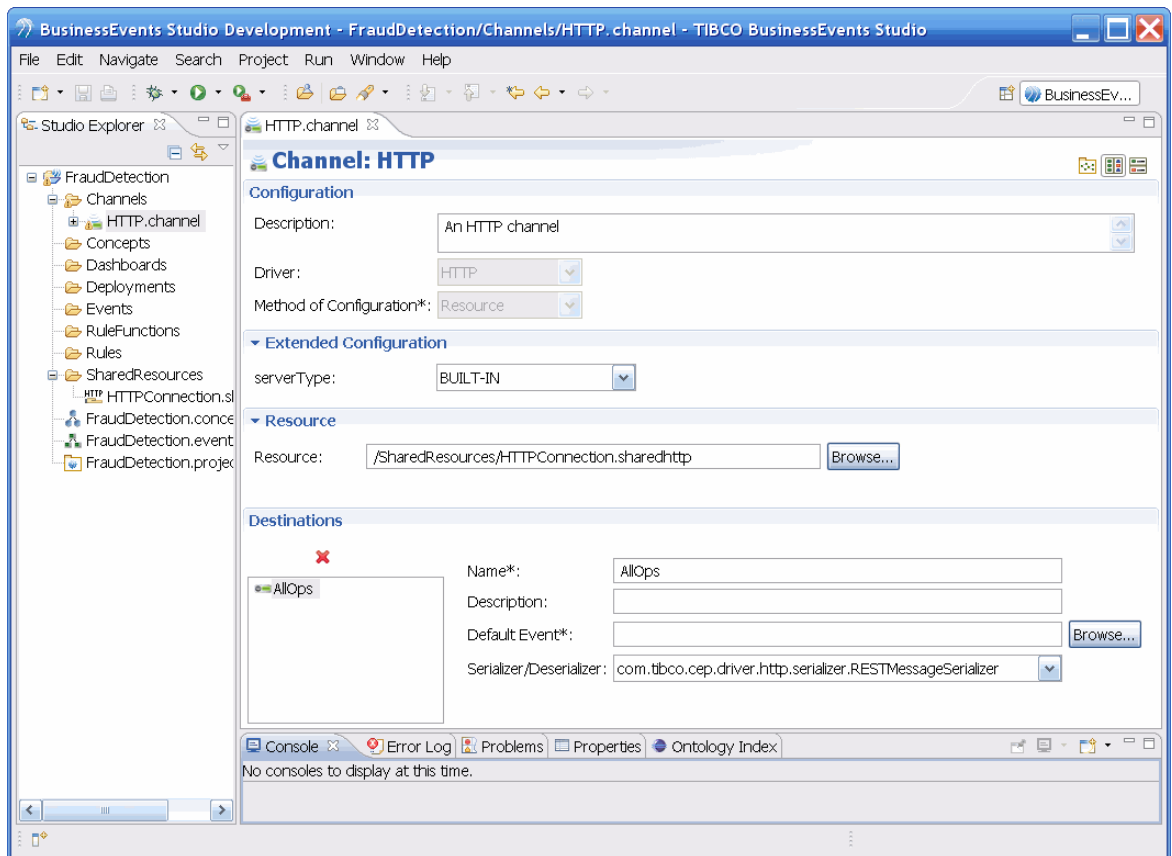
3. In the New HTTP Connection Wizard, name the connection **HTTPConnection** and click **Finish**. (In a real world situation you would probably give the connection a more meaningful name.) You see the HTTP Connection dialog.



Resource names and directory names in the path to a resource can't be any of the keywords or other words listed in Chapter 16, Rule Language Grammar in *TIBCO BusinessEvents Developer's Guide*, and they can't use spaces.

4. In the Host field, enter **localhost**.
5. In the Port field, enter **8104**. If that port is not free, use an available port in the 8000 range.
6. Save the resource (click the save button in the toolbar) and close it.

Task C Create an HTTP Channel and Destination



1. Right-click the **Channels** folder and select **New > Channel**.

2. You see the New Channel Wizard. In the Channel Name field, type **HTTP**.
 - a. In the Description field, type **An HTTP channel**.
 - b. In the Driver Type field, select **HTTP**.
 - c. Click **Finish**. You see the Channel editor



In BusinessEvents, you can't change any new project resource name after you click Finish in the new resource wizard. (You can change the description, however.) The resource name displays in the title of the resource editor, for example, Channel: HTTP.

3. In the Extended Configuration section, Server Type field, select **BUILT-IN**.
4. In the Resource field, browse to and select the HTTP connection resource you created in [Task B](#). Only valid shared resources for the current resource display.
5. In the Destinations section, click **Add** and name the destination **AllOps**. Leave all other fields set to their default values.



Provided Examples Requirements

- HTTP and AllOps are required names for examples. The `readme.html` uses the AllOps destination in its embedded forms.
- Adding a default event is not necessary for the provided examples because the `readme.html`, which starts an HTTP channel, specifies the event. However for other use cases, default events are convenient: all messages arriving at a destination are transformed to the destination's default event, unless the message specifies a different event. The default event in this channel is not required, but it does stop the warning sign from appearing! — the warning sign just lets you know when a destination does not have a default event.

6. Save and close the resource.

Summary and Next Steps

Now you have built a channel and a destination within that channel to listen for messages. The next step is to create some events.

Define the AccountOperation, CreateAccount, Debit, and Reply Events

In this task, you begin to build the project ontology by defining some events — or strictly speaking, event *types*. Before you define event types in a real-world project, you first examine the incoming and outgoing messages, as well as messages that you want to occur within the application, and configure each event type's characteristics accordingly. You can use inheritance (as demonstrated here) to simplify configuration.

BusinessEvents provides various kinds of events. Simple events are used in this tutorial to bring messages into the application. In addition you can use SOAP events, time events and advisory events, which you can learn about in the product documentation.

Learning Points

What is an event? The term *event* is overloaded: it means an activity that happens, and the definition of an object that represents the activity in BusinessEvents (an event type), and an instance of that event definition.

How are events (event instances) created? Simple event types are created at design time. Event instances are generally created using data in incoming messages. When a destination receives a message, it creates an event to hold the information from the message. Events from channels are automatically asserted into the Rete network, where their presence generally triggers rules (if all rule conditions are met).

Simple events can also be created by rules and rule functions. Events created this way are not asserted automatically because they could be outbound events, to be sent to a destination. You must explicitly assert such internally created events as needed.




Applications that Use Concurrency Features Require Use of Locking An important function of event preprocessors is locking, to ensure that multiple concurrent Rete networks (whether in the same agent or different agents in a cache cluster) do not work with the same object at the same time, or read an out of date version of the object. See Using Locks to Ensure Data Integrity Within and Across Agents in *TIBCO BusinessEvents Architect's Guide* to understand how to use locking correctly.

What is an event payload? Just as messages have properties and a message body, events can have properties and payloads. The payload is optional. It is used to hold more complex data, for example, SOAP messages. (The events in this example do not use a payload.)

What is a default destination? Outbound events of the same event type are often sent to the same destination. To simplify the process of sending those events, you can specify a default destination in the event type.



Why do the events have this warning sign?  Debit.event The yellow triangle with exclamation point means something may not be right. A red triangle means there is an error. The yellow triangle is shown for events with no default destination. In this case, however, no events are sent out through channels so no default is required.

What is a default event? The default event configured for a destination is used to hold information transferred from an incoming message, when no event type is specified in the message.

See Default Destinations and Default Events in *TIBCO BusinessEvents Architect's Guide* for more details. (The FraudDetection project does not use these features.)

Rules that apply to a parent type also apply to its child types Concept and event types use inheritance in a similar way to Java classes. Because *rules that apply to a parent type also apply to its child types*, it is generally not advisable to create many levels of inheritance. However it can be a useful technique. In this tutorial the AccountOperations event is the parent of both the CreateAccount and Debit events. You'll see why in a later section.

More Information

- Chapter 2, Channels and Events in *TIBCO BusinessEvents Architect's Guide* for overview and conceptual information.
- Chapter 7, Simple Events, Chapter 8, Time Events, and Chapter 9, Advisory Events in *TIBCO BusinessEvents Developer's Guide* for implementation details.

Task D Define the AccountOperations Event

This event is a parent to events that are used in the project. It has one property: AccountId. All its child events inherit this property, and extend the parent by adding more.

1. Right click the **Events** folder, and select **New > Simple Event**.
2. You see the New Simple Event Wizard. In the Simple Event Name field, type **AccountOperations**. In the Description field, type **Parent event for all account-related events**. Click **Finish**.
3. You see the Simple Event Editor. In the **Properties** section, click the **Add** button. Click in the cell under Name and type the name **AccountId**. It's a String property, and String is the default type.
4. Save and close the resource.

Task E Define the CreateAccount Event

Simple Event: CreateAccount

Configuration

Description: Triggers the CreateAccount rule to create an account

Inherits From: /Events/AccountOperations Browse...

Time to Live: 0 Seconds ▼

Default Destination: Browse...

Retry On Exception: ☒

Properties

Name	Type	Domain
AvgMonthlyBalance	double	
Balance	double	

For example purposes, the information needed to create an account is an ID, a balance, and a monthly average balance (for the fraud detection calculation). The ID is inherited from the AccountOperations event.

1. Right click the **Events** folder, and select **New > Simple Event**.
2. In the New Simple Event Wizard Filename field, type **CreateAccount**. In the Description field, type **Triggers the CreateAccount rule to create an account**. Click **Finish**.
3. In the Simple Event Editor Inherits From field, click Browse. In the upper section of the event picker, select the **Simple Event** event type, and in the lower section browse to and select the **AccountOperations** event. Click **OK**. The AccountOperations event is now the parent event for the CreateAccount one.
4. In the Default Destination field, click the browse button and in the Select Destination dialog, select **/Channels/HTTP.channel1/AllOps**. Click **OK**.
5. In the **Properties** section, add two properties:
 - **Balance**, of type **double**
 - **AvgMonthlyBalance**, of type **double**
6. Save and close the resource.

Task F Define the Debit Event

Following the procedure above, add the Debit event:

Field	Value
Name (Defined in the wizard)	Debit
Description (Defined in the wizard and editor)	Specifies an account (by inheritance) and a debit amount
Inherits From	AccountOperations
Properties	Amount (double)

Task G Define the Unsuspend Event

Following the procedure above, add the Unsuspend event:

Field	Value
Name (Defined in the wizard)	Unsuspend
Description (Defined in the wizard and editor)	Triggers the UnsuspendAccount rule to change the customer status from Suspended to Normal
Inherits From	AccountOperations
Properties	(No properties)

Summary and Next Steps

Next you will continue to configure the project ontology by defining a concept.

Define the Account Concept

In this task, you define the Account concept, which holds basic information about an account: an ID, a balance, an average monthly balance, and an account status. You also learn some useful information about concepts and how they are used.

Learning Points

What is a concept? A concept type is a definition of a set of properties that represent the data fields of an entity. Concept types are like Java classes, and concept instances are like Java objects.

How are concept instances created? Concept instances are created by rules and rule functions. Information from event properties or payloads is often used to create concept instances, but other information can be used, for example, the results of a query or a calculation.

What is a database concept? A BusinessEvents add-on product, TIBCO BusinessEvents Data Modeling, provides a feature that enables you to create concepts by importing them from a database. A set of functions enables you to update the database record to account for changes made in BusinessEvents. Unlike regular concepts, database concept instances are not asserted to the Rete network automatically, and they don't track history. (This add-on also provides a state modeler functionality.)

How can I persist concept instances? Instances of concepts (and events) are also known as "facts" and "entities." They can be persisted in various ways, generally using a cache and backing store, as determined by the business need. Later tutorials explain these features.

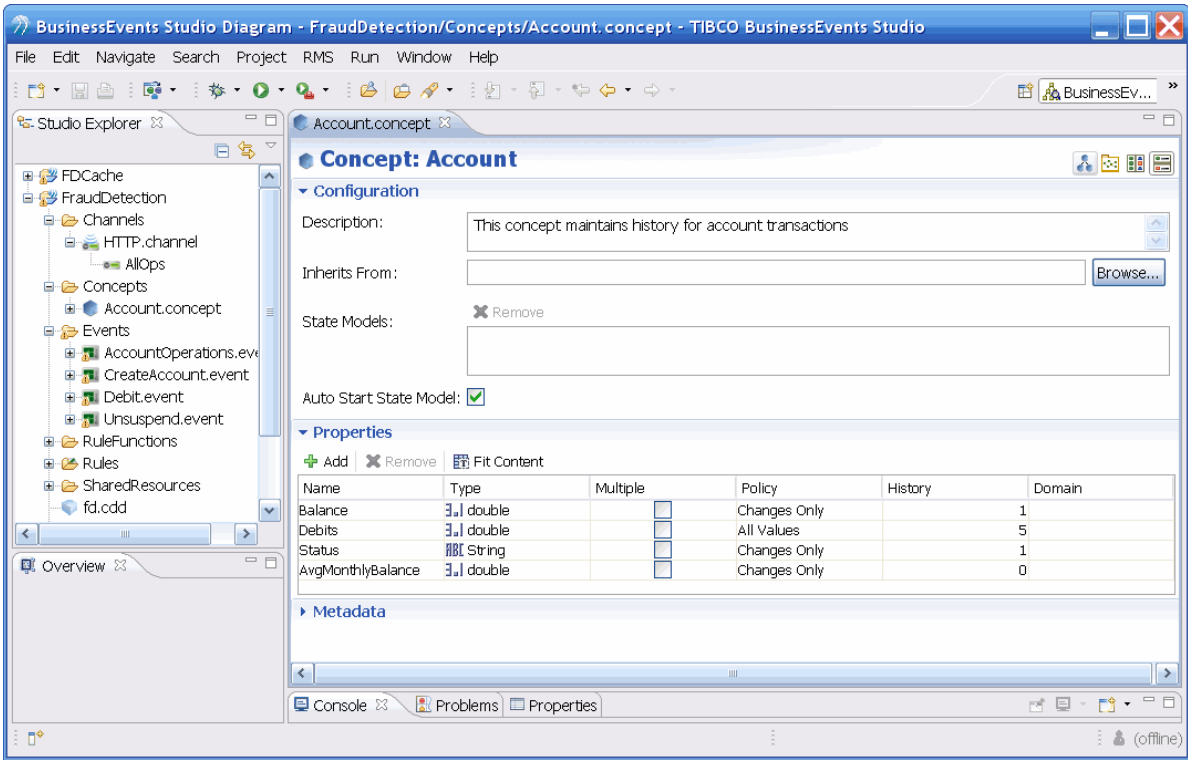
How is history tracked? When the History setting for a concept property is 0 (zero) the current value is stored without a date-time stamp. When the history setting is 1, the current value is stored, along with the date and time the value was added or changed. When the history value is greater than 1, BusinessEvents tracks changes to property values up to the specified number (using a ring buffer). The Policy setting additionally determines what values are recorded, all values or only changes to the prior value.

You'll set the Debits property history, to track "All Values," that is, BusinessEvents records the value of the property every time an action sets the value, even if the new value is the same as the old value — a person can debit the account twice by the same amount. For a property such as "address" you might want to track only changes to the value.

More Information

- Chapter 3, Concepts in *TIBCO BusinessEvents Architect's Guide*.
- Chapter 10, Concepts in *TIBCO BusinessEvents Developer's Guide*.
- *TIBCO BusinessEvents Data Modeling Developer's Guide*

Task H Define the Account Concept



- 1. Right click the **Concepts** folder, and select **New > Concept**.
- 2. You see the New Concept Wizard. In the Filename field, type **Account**. In the Description field, type **This concept maintains history for account transactions**. Click **Finish**.
- 3. In the Account Concept Editor Properties section, add the following properties:

Name	Type	Policy	History
Balance	double	Changes Only	1
Debits	double	All Values	5
Status	String	Changes Only	1
AvgMonthlyBalance	double	Changes Only	0

The Multiple field is used to define an array property. In this release, domains are used only with TIBCO BusinessEvents Decision Manager, a BusinessEvents add-on product.

You may wonder where the account ID from the incoming event will be stored. It will go into the concept's `extId` attribute.



Concept Attributes and Concept Relationships

Attributes Concepts, events, and scorecards have some built-in attributes, in addition to the properties you define here. The attribute `extId`, referenced as `@extId`, will hold the account ID.

Concept Relationships Concepts can have containment and reference relationships with other concepts. You set these up as concept properties, and define the kind of relationship by selecting an appropriate data type for the property. For example, a car concept can contain Wheel concepts, and can have a reference relationship to a Dealership concept.

Inheritance Concepts can also inherit from other concepts (and events can inherit from other events). Inheritance is a programming relationship and not an ontology relationship. It enables you to extend a base type for more efficient project management. See [Define the AccountOperation, CreateAccount, Debit, and Reply Events on page 13](#) for important information about inheritance.

4. Save and close the resource.

Summary and Next Steps

You have defined a concept type to hold information about bank accounts. The last step in building the ontology of your project is to set up a scorecard to hold fraud detection criteria that are used in rules.

Add the FraudCriteria Scorecard

In this task, you finish building the project ontology by creating a scorecard. The `FraudCriteria` scorecard will store the criteria used to determine fraud, not any specific data about customer accounts. In this example, you will use this scorecard in rules.

Learning Points **What is a scorecard?** A scorecard is a special type of concept. A scorecard serves as a static variable. You can use a scorecard resource to track key performance indicators or any other information. Unlike concepts, there is only one instance of a scorecard. You create the scorecard at design time. Its values can be viewed and updated using rules.

It is more accurate to say there is one instance of a scorecard per inference agent. This tutorial uses one inference agent. However, in the next tutorial you will deploy multiple agents, and each has its own instance of the scorecard. This enables scorecards to be used for local purposes and minimizes contention between the agents. Do not use scorecards as a mechanism to share data between multiple agents.

More Information • Chapter 11, Scorecards in *TIBCO BusinessEvents Developer's Guide*.

Task I Create the FraudCriteria Scorecard

There is only one scorecard in this project, so we don't need a folder for it. You will create it in the project root.

1. Select the `FraudDetection` root folder and press **Ctrl+N**. In the Select a Wizard dialog, expand **TIBCO BusinessEvents** and click select **Scorecard**.

This is how you add resources that are less commonly used.

2. Name the scorecard **FraudCriteria**. Add the description **Stores the criteria used to determine fraud**, and click **Finish**.
3. In the `FraudCriteria` Scorecard editor, add the following properties:

Name	Type	Policy	History
interval	long	Changes Only	0
num_txns	int	Changes Only	0
debits_percent	double	Changes Only	0

4. Save and close the resource.

**Summary and
Next Steps**

You have now set up the ontology for the project, that is, the definitions of all the events, concepts, and scorecards that are needed to store information (facts) about possible fraud detection.

Next, you will configure a rule function that sets values for the scorecard, and one that acts as an *event preprocessor* — a term you will learn about in that section.

Add the InitializeScorecard Rule Function

In this task, you configure a rule function that initializes values for the `FraudCriteria` scorecard. This rule function is used at system startup. (You'll configure that connection later.)

Learning Points

What is a rule function? A rule function is a function you write in the BusinessEvents rule language.

How are rule functions created and used? You write rule functions using the rule editor. You can use rule functions in rules and other rule functions, in event preprocessors, and as startup or shutdown functions for an agent.

What other types of functions are there? BusinessEvents provides a large library of functions for various purposes. In addition, an ontology function is automatically created for each concept and event in your project. These are constructor functions. Another type of ontology function enables you to create and schedule a time event. You can also add custom Java functions.

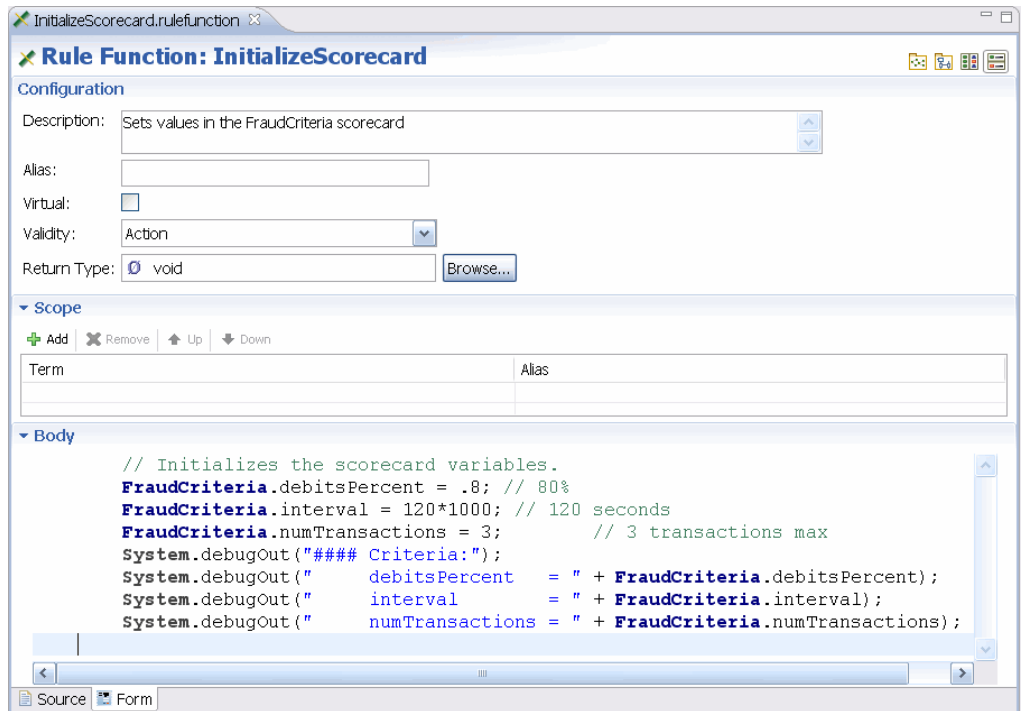
More Information

- Chapter 4, Rules and Functions in *TIBCO BusinessEvents Architect's Guide*.
- In *TIBCO BusinessEvents Developer's Guide*, the following chapters:
 - Chapter 14, Rules and Rule Functions
 - Chapter 15, Functions
 - Chapter 16, Rule Language Grammar
 - Chapter 17, Rule Language Datatypes
 - Chapter 18, Mapping and Transforming Data
 - Chapter 19, XPath Formula Builder

Task J Add the InitializeScorecard Rule Function

1. Right click the **RuleFunctions** folder, and select **New > Rule Function**
2. You see the New Rule Function Wizard. In the Filename field, type **InitializeScorecard**. In the Description field, type **Sets values in the FraudCriteria scorecard**. Click **Finish**.

You can work in the Source view or the Form view, according to your preference. The tutorial uses the Form view. In the lower area, click the Form tab to switch to the Form view.



Rule Function Editor Preference To set the default mode, go to Window > Preferences > BusinessEvents > Rules and check or uncheck the following checkbox as desired: **Initially show 'Form' tab in Rule Function Editor.**

3. Leave the Scope area empty, because this function is used at startup. In the Body area, add the following lines to provide values to the FraudCriteria scorecard (and to comment your code):

```
//Intialize scorecard variables
FraudCriteria.debits_percent = .8;
FraudCriteria.interval      = 120*1000; /* 120 seconds */
FraudCriteria.num_txns = 3;
```

Notice that when you type the period after `FraudCriteria`, a list of its properties appears so you can select a property.

4. Save and close the resource.

Summary and Next Steps

You've set up a startup rule function. Next you'll set up an event preprocessor rule function.

Add the PreProcessor Rule Function

In this task, you configure a rule function that replies to the request received from the HTTP channel. HTTP is a request-reply protocol, and this step is required so that the HTTP server is ready to process the next request from the `readme.html` form. This rule function executes when an event is received. (You'll configure that connection later.)

Learning Points

What is an event preprocessor? An event preprocessor is a rule function that processes incoming messages before BusinessEvents transforms them into events. In this case the preprocessor is used to send a response to the HTTP server. In real-world applications, however, a preprocessor might filter the messages so that only certain ones are used as events, and it might do other event enrichment actions. Preprocessors are multi-threaded and you can choose from various threading and queue options, as appropriate to handle the work load. By default the threading uses the system-wide shared queue and threads. See the topic *Event Preprocessors* in *TIBCO BusinessEvents Architect's Guide*.

More Information

- Event Preprocessors in *TIBCO BusinessEvents Architect's Guide*.

Task K Add the PreProcessor Rule Function

1. Right click the **RuleFunctions** folder, and select **New > Rule Function**
2. You see the New Rule Function Wizard. In the Filename field, type **PreProcessor**. In the Description field, type **Closes requests from the HTTP server**. Click **Finish**.
3. Click the **Form** tab at the bottom of the editor.
4. In the Scope section, click **Add**. You see the Select Rule Function Scope Arguments dialog.
5. In the Types area (at the top), select **Event**. Click **OK**.

In the Select Resource area, you could select a specific event type in the project. However, here we want any event to be in the scope of this rule function, not one specific event type.

6. In the Alias area replace the default alias (e) with **Request**.
7. In the Body area, type: **Event .** and notice that when you type the period (.) you see a list of all catalog functions in the Event category. Use the down

arrow to scroll down the list of functions and stop at **replyEvent**. Its tooltip displays. Documentation for all catalog functions is provided in tooltips.

Function:	<i>Event.replyEvent</i>	
Signature:	<code>boolean replyEvent(SimpleEvent request, SimpleEvent reply)</code>	
Synopsis:	Replies with a reply SimpleEvent to a request SimpleEvent. If a reply destination on the request SimpleEvent is specified, it will be used to send the reply SimpleEvent, else no action is taken.	
Parameters:		
	<u>Name</u>	<u>Type</u> <u>Description</u>
	<i>request</i>	SimpleEvent The original request SimpleEvent.
	<i>reply</i>	SimpleEvent The Reply SimpleEvent.
Returns:	boolean true if the SimpleEvent is sent; false otherwise.	
Cautions:		
Example:		

The tooltips are also reproduced in the HTML version of the product documentation, in the Online References area.

8. Click the **replyEvent** function to select it. Now the body looks like this:

```
Event.replyEvent(
```

As you can see, the rule function arguments are two events, a request event and a reply event.

9. To specify the request event, type **request**, the alias for the scope argument you added in [step 5](#).
10. To specify the reply event, just type **request** again. The reply event can be any event in this case, so we can simply reply with the request event.

Summary and Next Steps

You have configured a rule function that will send a reply to requests sent by the HTTP server (through the HTTP channel). Next you will configure rules that take action on assertion of CreateAccount and Debit events, depending on various conditions.

Add BadCreateAccount and CreateAccount Rules

In this task you create two rules, one called `CreateAccount` and one called `BadCreateAccount`. Both rules can fire when a `CreateAccount` event is asserted into the Rete network. However, the `BadCreateAccount` rule has a higher priority, so it will fire before the `CreateAccount` rule.

The `BadCreateAccount` rule checks whether the account ID provided in the `CreateAccount` event matches the account ID of any `Account` instance already in the Rete network. One of the two following situations must occur:

- A matching ID exists in the Rete network: the `BadCreateAccount` rule prints a message to the console, and "consumes" — that is, deletes — the `CreateAccount` event. Because that event is consumed, the `CreateAccount` rule cannot fire.
- No matching ID exists in the Rete network: the `BadCreateAccount` rule does nothing, and then the `CreateAccount` rule fires, and creates the `Account` concept instance.

Learning Points

What are rules and how are they created? Rules define actions to take when certain conditions are met. Rules are written in the `BusinessEvents` rule language, which is similar to the Java language. Rules are declarative and are generally narrow in scope. A rule has three parts: the declaration (`declare`), the conditions (`when`), and the actions (`then`). As with rule functions, you can work in a source view, which displays the Java-like code, or in a form view.

How are rules used at runtime? The rule engine checks all changes and additions to the Rete network and evaluates or reevaluates rules, using their declaration and conditions, as needed. Eligible rules are added to the *rule agenda*.

What is the rule agenda A rule fires when it is at the top of the agenda. The engine determines the order of firing using rule declarations and conditions, and each rule's priority and rank (if these features are used). As the contents of the Rete network change, the engine reevaluates rules and removes any that are no longer eligible to fire. See *Understanding Conflict Resolution and Run to Completion Cycles* in *TIBCO BusinessEvents Architect's Guide* for details.

How can you prioritize rule execution? The Priority setting is used by the runtime engine when determining the order in which rules are fired. Those with a number closer to one fire first. Within a set of rules that has the same priority, a ranking feature enables you to determine which fire before others. It uses a callback rule function that enables you to specify business logic to establish the rank. When there is no reason to force rules to execute in a particular order, leave the Priority and Rank fields set to the default and let the runtime engine determine rule order.

More Information

- See all references provided for [Add the InitializeScorecard Rule Function on page 22](#)
- Chapter 5, Run-time Inferencing Behavior in *TIBCO BusinessEvents Architect's Guide*.

Task L Add the BadCreateAccount Rule

1. Right click the **Rules** folder, and select **New > Rule**.
2. You see the New Rule Wizard. In the Filename field, type **BadCreateAccount**. In the Description field, type **Checks for an existing account with the specified ID**. Click **Finish**.
3. In the Form tab, set the Priority field to **3**. It's higher priority than 5, the default. You'll set the CreateAccount rule to priority 5, so that the BadCreateAccount rule always fires before CreateAccount rule.
4. Expand the Declaration section as needed so you can see empty rows below the headings Term and Alias.

Each of the sections can be expanded and contracted as needed.

5. Drag the Account concept from the Studio Explorer tree into the first empty row in the Declaration section. You see the project path of the Account concept in the Term column, and account in the Alias column.
6. Similarly, drag the CreateAccount event into the next available row.

Declaration The Declaration provides the scope of the rule. It lists all the entity types to be used in the rule, and their aliases. By default the alias is set to the entity name in lower case letters. You can change it as desired.

7. In the Conditions panel, type the following:

```
//Checks whether the extId of an Account instance in working memory
//matches the incoming event's account ID
```

```
account@extId == createaccount.AccountId;
```

Notice that when you type the At sign (@), a pick list of concept attributes appears. Attributes are built-in. You can't add or remove attributes. The `id` attribute value is set internally. However you can set the external ID, `extId`.

8. In the Actions panel, just below the Conditions panel, type these statements:

```
System.debugOut("#### Account already exists: " + createaccount.AccountID;
Event.consumeEvent(createaccount);
```

These actions are done only if the conditions are met. If not, then the next rule in the agenda fires — and that is likely to be the `CreateAccount` rule, which you'll define next.

You are also consuming the event, so it can't trigger any other rules.

9. Save and close the resource.

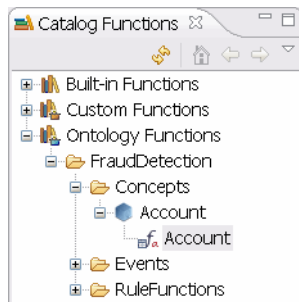
Task M Add the CreateAccount Rule

1. Right click the **Rules** folder again, and select **New > Rule**.
2. In the New Rule Wizard Filename field, type **CreateAccount**. It doesn't really need a description does it? Click **Finish**.
3. In the Form tab, set the Priority field to **5**. (This is a lower priority than the `BadCreateAccount` rule.)
4. Drag the `CreateAccount` event from the Studio Explorer tree into the first empty row in the Declaration section.

This rule has no conditions — the `BadCreateAccount` means there is no need. You could have combined the two rules into one. There are many ways to write rules for a project. You have to use your judgment.

In the Actions panel, you'll use an ontology function to create the Account concept instance. You can also get to ontology functions using the function catalog.

5. From the top menu select **Window > Show View > Other > TIBCO BusinessEvents > Catalog Functions**. The Catalog Functions view displays on the right (unless your Eclipse IDE is configured differently — it could display along the bottom, for example).
6. Below each concept, event, and rule function is its ontology function. Expand **Ontology Functions > FraudDetection > Concepts > Account > Account**:



7. Drag the Account function into the Actions section. You see its signature:


```
Concepts.Account.Account(/*extId String *//*Balance double *//*Debits double
*//*Status String *//*AvgMonthlyBalance double */)
```

8. Configure the function to create the Account concept, and type the rest of the actions as follows. You can double click the tab to expand the rule editor, to make rule writing easier.
-

```
Concepts.Account.Account(createaccount.AccountID/*extId String */,
    createaccount.Balance/*Balance double */,
    0/*Debits double */,
    "Normal"/*Status String */,
    createaccount.AvgMonthlyBalance/*AvgMonthlyBalance double */);

Event.consumeEvent(createaccount);

System.debugOut("#### Created account " + createaccount.AccountID);
```

9. Save and close the resource.

Note that in this case it is not really necessary to consume the event. No other rules have this event in their scope. Events with time to live (TTL) set to zero (0) are consumed at the end of an RTC. However it does no harm and makes the rules more consistent, reducing chances of error.

Add ApplyDebit, BadApplyDebit, and CheckNegativeBalance Rules

The rules you add in this section work together as follows:

When a Debit event is asserted into the Rete network, BusinessEvents checks rules with the Debit event in their scope.

- The ApplyDebit rule has the Debit event in its scope, and also an account ID. It's priority 1 so it will execute before any other lower priority rule. The engine checks the rule conditions.
 - If the Rete network also contains an Account instance whose ID matches the ID in the debit event, the rule executes. If the account is suspended, a message to that effect displays in the console. Otherwise, the account is debited. A message displays in the console to this effect.
 - If the Rete network does not contain a matching Account instance, then other rules in the agenda — those that have debit events in their scope — are eligible to execute. BadApplyDebit is the only rule with the Debit event in its scope, so it is eligible and it executes. It sends a message to the console that no matching account is found.
- The CheckNegativeBalance rule has an Account concept in its scope. When an account is debited the Account concept is changed, and so the CheckNegativeBalance rule becomes "newly true." The effect is similar to assertion of a new entity into the Rete Network. If the debit has made the account balance negative, this rule sets the status to Suspended.

Learning Points

It's important to understand how conflict resolution and run to completion (RTC) cycles work. If you understand what triggers rules to execute, and why a rule may not execute, you can design rules more effectively. For a reminder, carefully reread Understanding Conflict Resolution and Run to Completion Cycles in *TIBCO BusinessEvents Architect's Guide*.

More Information

- See all references provided for [Add the InitializeScorecard Rule Function on page 22](#)
- Chapter 5, Run-time Inferencing Behavior in *TIBCO BusinessEvents Architect's Guide*.

Task N Add ApplyDebit, BadApplyDebit, and CheckNegativeBalance Rules

If you have completed [Task L, Add the BadCreateAccount Rule](#) and [Task M, Add the CreateAccount Rule](#), you will be familiar with adding rules. This section shows the source code for the three rules you will add. As an extra hint, screenshot of the ApplyDebit rule is also shown below

ApplyDebit Rule Form View

The main purpose for showing the form view is so you can compare it with the source view when creating your own rule.

Rule: ApplyDebit

Configuration

Description: Debits the matching account by the specified amount

Priority: 1 (Highest)

Rank: Browse...

Forward Chain: ☒

Declaration

Term	Alias
/Events/Debit	debit
/Concepts/Account	account

Conditions

```
//Checks whether the extId of an Account instance in working memory
//matches the incoming event's account ID
account@extId == debit.AccountID;
```

Actions

```
//If Account Status is not Suspended, debits the account
if (account.Status != "Suspended") {
    account.Debits=debit.Amount;
    System.debugOut("#### Debiting account " +account@extId+ " by $" +debit.Amount);
    account.Balance=account.Balance - debit.Amount;
    System.debugOut("#### New balance: $" + account.Balance);
}
else {
    System.debugOut("#### Cannot debit the suspended account " +account@extId );
}

RuleFunctions.CloseRequest(debit);
Event.consumeEvent(debit);
```

ApplyDebit Rule Source View Code

```

/**
 * @description Debits the matching account by the specified amount
 * @author
 */
rule Rules.ProcessDebits.ApplyDebit {
    attribute {
        priority = 1;
        forwardChain = true;
    }
    declare {
        Events.Debit debit;
        Concepts.Account account;
    }
    when {
        //Checks whether the extId of an Account instance in working memory
        //matches the incoming event's account ID
        account@extId == debit.AccountId;
    }
    then {
        //If Account Status is not Suspended, debits the account
        if (account.Status != "Suspended") {
            account.Debits=debit.Amount;
            System.debugOut(
                "#### Debiting account " +account@extId+ " by $" +debit.Amount);
            account.Balance=account.Balance - debit.Amount;
            System.debugOut("#### New balance: $" + account.Balance);
        }
        else {
            System.debugOut(
                "#### Cannot debit the suspended account " +account@extId );
        }
    }
    Event.consumeEvent(debit);
}
}

```

BadApplyDebit Rule Source View Code

```
/**
 * @description
 * @author
 */
rule Rules.BadApplyDebit {
  attribute {
    priority = 5;
    forwardChain = true;
  }
  declare {
    Events.Debit debit;
  }
  when {

  }
  then {
    System.debugOut(
      "#### Debit not applied, account not found: " + debit.AccountId);
  }
}
```

CheckNegativeBalance Rule Source View Code

```
/**
 * @description
 * @author
 */
rule Rules.CheckNegativeBalance {
  attribute {
    priority = 5;
    forwardChain = true;
  }
  declare {
    Concepts.Account account;
  }
  when {
    //Checks that the balance is less than zero
    account.Balance < 0;
    //Checks that Account status is not set to Suspended
    account.Status!="Suspended";
  }
  then {
    account.Status="Suspended";
    System.debugOut(
      "#### Account ID "+account.extId+" STATUS set to Suspended. Balance"
      +account.Balance+" is less than zero");
  }
}
```

Add the FraudDetection Rule and Unsuspend Account Rule

You are an experienced rule builder now! Add the `FraudDetection` rule, typing in the Source view or Form view, according to your preference. The source is shown below.

Most of the code is in the conditions. The first condition checks whether the number of debits in the specified interval prior to the current time is greater than the specified number of debits. The interval and the number of debits are set in the `FraudCriteria` scorecard.

The second condition checks whether the sum of all debits in the verification interval is greater than the specified percentage of the account average monthly balance. The specified percentage is set in the `FraudCriteria` scorecard. The average monthly balance, for the purposes of this tutorial, is set in the Account instance created by the `CreateAccount` rule.

You will also add a rule called `UnsuspendAccount`. This is a convenience rule that allows you to change a customer status from `Suspended` to `Normal` at run-time.

Learning Points

How are conditions processed? All conditions in a rule must be met, before the action is done. That is, each condition is joined by an implied AND operator.

In what order are conditions evaluated? To learn more the effect of filters, equivalent join conditions, and non-equivalent join conditions on the efficiency of a rule, see *Order of Evaluation of Rule Conditions* in *TIBCO BusinessEvents Architect's Guide*. Understanding these points helps you design an efficient project.

How can I learn about all these catalog functions? `BusinessEvents` provides hundreds of catalog functions for use in rules and rule functions. You can use functions you already know about by typing the beginning of the name and then using the completion hints that appear. To learn about more functions, you can open the Catalog Functions view and browse. To see the tooltip for a function, hover the mouse over the function name. You can then drag a function into the editor, as you did in [Task M, Add the CreateAccount Rule](#). As a reminder, here's how to open the Catalog Functions view: **Window > Show View > Other > TIBCO BusinessEvents > Catalog Functions**. The tooltips are also available in HTML form, in the Online References section of the HTML documentation for the product.

More Information

- See all references provided for [Add the InitializeScorecard Rule Function on page 22](#)
- Chapter 5, Run-time Inferencing Behavior in *TIBCO BusinessEvents Architect's Guide*.

Task O Add the FraudDetection Rule

```

/**
 * @description
 * @author
 */
rule Rules.FraudDetection {
    attribute {
        priority = 5;
        forwardChain = true;
    }
    declare {
        Concepts.Account account;
    }
    when {
        //1. Checks the number of debits in the verification interval
        Temporal.History.howMany(account.Debits,
            DateTime.getTimeInMillis(DateTime.now())-FraudCriteria.interval,
            DateTime.getTimeInMillis(DateTime.now()),
            true)
        > FraudCriteria.num_txns;

        //2. Checks the percentage of the average balance that was
        // debited in the verification interval
        Temporal.Numeric.addAllHistoryDouble(account.Debits,
            DateTime.getTimeInMillis(DateTime.now())-FraudCriteria.interval)
        > FraudCriteria.debits_percent*account.AvgMonthlyBalance;

        //Check whether Account status is not set to Suspended
        account.Status!="Suspended";
    }
    then {
        account.Status="Suspended";
        System.debugOut("#### Account ID "+account.extId+" STATUS set to Suspended.
        Fraud suspected.");
    }
}

```

Task P Add the UnsuspendAccount Rule

```

/**
 * @description Closes requests from the HTTP server
 */
void rulefunction RuleFunctions.PreProcessor {

    attribute {
        validity = ACTION;
    }
    scope {
        Event request;
    }
}

```

```
}  
body {  
    // Replies to the request event, in order to close the HTTP request.  
    // To keep it simple, uses the request event as the response.  
    Event.replyEvent(request, request);  
}  
  
}
```

**Summary and
Next Steps**

Congratulations! You have now configured the project's ontology and rules. Now you are ready to configure the Cluster Deployment Descriptor and build the archive for deployment. But before you do, it's wise to validate and analyze the project, and look at it in the Project diagram.

Analyze and Validate the Project

In this task, you check the project for errors. If you have configured it correctly, you'll see warnings but no errors. Warnings indicate potential issues, such as absence of default events. It's always worth checking them. In the `FraudDetection` project, the warnings do not indicate any actual issues.

Learning Points

How can I validate that there are no basic errors in my code? To perform checks that don't take project logic into account, use `Project > Validate`.

What if there are problems in my code? You can debug it. `BusinessEvents` builds on the Eclipse debugger features to provide additional checks. Developers familiar with the Eclipse debugger will find it intuitive to use.

How can I analyze that the logic of my project is OK? For this kind of checking you use `Project Analyzer` or the `Element Diagram` or both. They work together to provide insights into your project. `Project Analyzer` is a document generated in the `Problems` view area. `Element Diagram` provides a visualization of the whole project or of selected project elements. It also shows all dependencies in a project. `Project Analyzer` and `Element Diagram` can be configured to run separately or together, using preferences.

How can I visualize different aspects of a project? `BusinessEvents` provides diagrams similar to UML sequence diagrams, class diagrams, and dependency diagrams. They are modified a little from the strict UML to be more useful in the context of `BusinessEvents` Monitoring and Management.

How can I change the names of project resources, or move them around? Use the refactoring features. They ensure that changes you make are reflected throughout the project (with certain limitations).

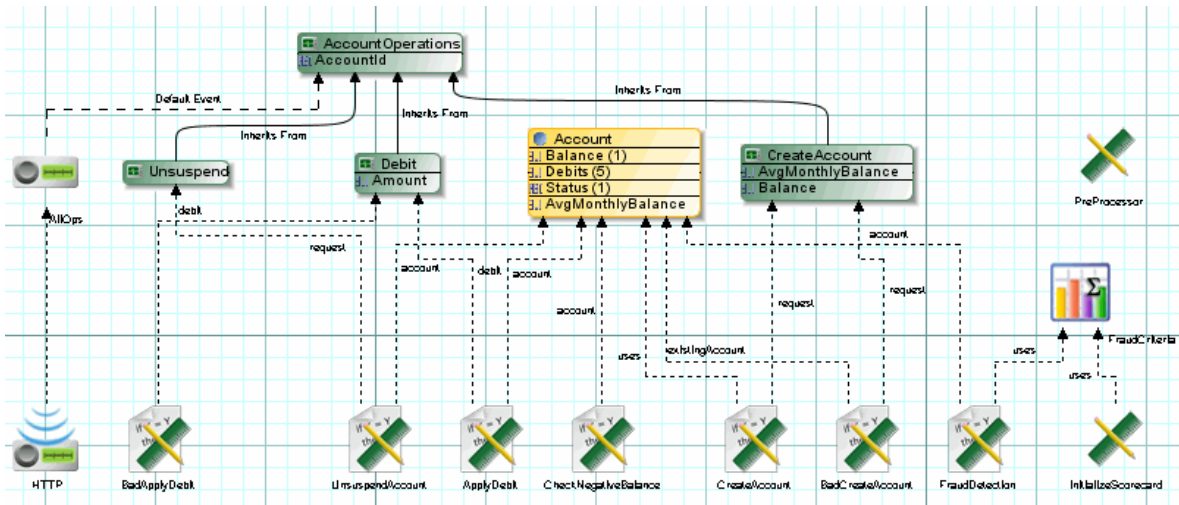
More Information

In *TIBCO BusinessEvents Developer's Guide*, see these sections:

- Chapter 3, Element Refactoring Operations
- Chapter 23, Debugger in *TIBCO BusinessEvents Developer's Guide*
- Chapter 24, Diagrams, especially `Project Analyzer` and `Selected Entity Project Diagrams`, in

Task Q Analyze and Validate the Project

- 1. In Studio Explorer, highlight the project name, then from the top menus select **Project > View**. You see the project visualization in the main editor window.



You can see the dependencies and relationships between the project resources. You can jump to a resource editor by clicking the resource’s icon.

- 2. Click the Problems tab to see the project analysis.

The Type column shows the type of check that resulted in the warning. You can sort by any column. Here are some warnings taken during project development. They are sorted by Resource and some columns are sized smaller to make others more visible.

Problems				
14 warnings, 0 others				
Option	Resource	Path	Location	Type
Warnings (14 items)				
Event /Events/AccountOperations does not have any destination configured for it	AccountOperations.event	FraudD...	Unkno...	Project Analyzer Problems
A default destination is not specified	AccountOperations.event	FraudD...	Unkno...	Resource Validation Problem
Rule /Rules/ApplyDebit may never fire (it has unused event in its scope)	ApplyDebit.rule	FraudD...	Unkno...	Project Analyzer Problems
Rule /Rules/BadApplyDebit may never fire (it has unused event in its scope)	BadApplyDebit.rule	FraudD...	Unkno...	Project Analyzer Problems
Rule /Rules/BadCreateAccount may never fire (it has unused event in its scope)	BadCreateAccount.rule	FraudD...	Unkno...	Project Analyzer Problems
Event /Events/CreateAccount does not have any destination configured for it	CreateAccount.event	FraudD...	Unkno...	Project Analyzer Problems
A default destination is not specified	CreateAccount.event	FraudD...	Unkno...	Resource Validation Problem
Rule /Rules/CreateAccount may never fire (it has unused event in its scope)	CreateAccount.rule	FraudD...	Unkno...	Project Analyzer Problems
Event /Events/Debit does not have any destination configured for it	Debit.event	FraudD...	Unkno...	Project Analyzer Problems
A default destination is not specified	Debit.event	FraudD...	Unkno...	Resource Validation Problem
The Event URI in the Destination "AllOps" of Channel "HTTP" is empty.	HTTP.channel	FraudD...	Unkno...	Resource Validation Problem
Destination AllOps in Channel /Channels/HTTP does not have a default event	HTTP.channel	FraudD...	Unkno...	Project Analyzer Problems
Rule Function /RuleFunctions/InitializeScorecard is never used	InitializeScorecard.rulefun...	FraudD...	Unkno...	Project Analyzer Problems
A default destination is not specified	Reply.event	FraudD...	Unkno...	Resource Validation Problem

3. Close the `FraudDetection.projectview` tab. Diagrams are not saved. They are generated when needed.

**Summary and
Next Steps**

Now you are ready to configure the Cluster Deployment Descriptor (CDD) and build the archive for deployment.

Add a Cluster Deployment Descriptor and Build the EAR File

To deploy a project you need a CDD file and an EAR file. The CDD is not included in the project EAR. This means you can reconfigure a project's deployment configuration without having to rebuild the EAR.

Learning Points

What is an EAR The Enterprise Archive or EAR file contains details for all the resources in a project, and project global variables.

What is a CDD The project's deployment configuration is defined in an XML file called the Cluster Deployment Descriptor, or CDD. You edit this file using the BusinessEvents Studio Cluster Deployment Descriptor editor.

How do I set up preprocessors and startup and shutdown rule functions? The CDD is where you configure rule functions to act as event preprocessors, startup rule functions, or shutdown rule functions! Only rule functions whose Validity setting is Action are valid for these uses. Such rule functions cannot require anything to be in their scope, because they execute outside of the context of the Rete network and BusinessEvents project resources.

What is an object manager? Object management refers to how BusinessEvents manages the objects generated within the BusinessEvents application at runtime. For some applications, the objects are more important than for others. This tutorial focuses on the basics, so it uses In Memory, the simplest object manager. Objects are kept in memory only, and are lost when the engine stops.

What are agents An agent does certain work within the engine. With the cache object manager, different types of agents do different work. But for a simple In Memory project, you use only one agent of one type, and that is the inference agent.

What is a processing unit? A processing unit is deployed as a BusinessEvents Studio Explorer engine. It has one or more agents and it runs in one JVM.

More Information

In *TIBCO BusinessEvents Administration*, see these chapters:

- Chapter 2, CDD Configuration Procedures
- Chapter 3, Cluster Deployment Descriptor Reference

Task R Add a CDD

1. In Studio Explorer, right click the project name, *FraudDetection*, and select **New > Cluster Deployment Descriptor**. You see the New Cluster Configuration Wizard.

You can create multiple CDD files for a project and at deploy time use the one that has the configuration you want to use.

2. In the File name field, type **fd** and click **Next**.

Unlike other project resources, you can change the name later as desired. Short names are easier to type when starting the engine at the command line.

3. At the Template Selection page, select In Memory from the Object Management Type drop-down list. Then click **Finish**.

You see the CDD editor. For an In Memory project, very little deployment configuration is required.

4. In the Agent Classes tab, expand the default agent class, which is called inference-class. (Agent classes can be used in cache object manager to define different types of agents.)

When configuring an agent class, you can select a subset of the project rules, select and configure various destinations, and select rule functions that execute at engine startup and shut down, as needed.

5. Select **Destination Collections** and click **Add**. In the Select Destinations dialog, select **/Channels/HTTP/AllOps** and click **OK**.

In the Configuration panel, a generated ID for the destination appears, along with various configuration options.

In the Preprocessor field, select **/RuleFunctions/PreProcessor**.

6. In the Threading Model field select **Caller**. For the HTTP channel, this threading model is required.
7. Select **Startup Functions** and click **Add**. In the Select Destinations dialog, select **/Channels/HTTP/AllOps** and click **OK**. When the engine starts, this rule function executes and initializes the scorecard values.
8. Save and close the CDD.

Task S Build the EAR File

You must build the EAR file outside the project tree, or it will be recursively included in the next EAR file you build!

1. In Studio Explorer, highlight the project name, then from the top menus select **Project > Build Enterprise Archive**.

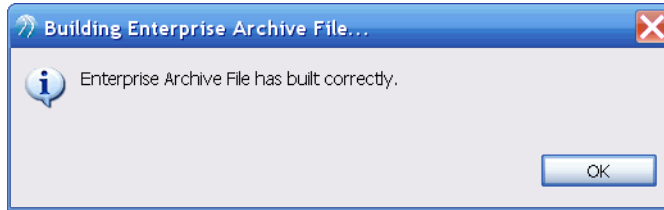
If you see a message asking you to save all project resources, click Yes. It means an unsaved resource editor is open.

At the Build Enterprise Archive dialog, change the name to **fd**. (Short names are easier to type when starting the engine at the command line.)

2. In the File Location field, browse to and select the directory above the project directory. To build the EAR in the provided example location, choose

BE_HOME/examples/FraudDetection/FD.ear. Replace *BE_HOME* with your actual value.

3. Press Apply, then press OK. You see messages as the EAR file builds, then you see a message that the EAR file has built correctly:



**Summary and
Next Step**

You are ready to deploy the FraudDetection project!

Start the Engine and Run the Application

You can simply open the example `readme.html` file and follow instructions there. In order to send events into the engines, you must enter information into the forms provided in the `readme`.

To Start the Application at the Command Line

For your information, here is how to start the engine, using the default locations specified for this example. If you have built the example in a different location, please adapt the instructions:

Open a command window and at the command prompt, start the BusinessEvents engine using the following command. Substitute your actual value for `BE_HOME`:

```
cd BE_HOME/examples/FraudDetection
BE_HOME/bin/be-engine --propFile BE_HOME\bin\be-engine.tra -u default -c
FraudDetection/fd.cdd fd.ear
```

The format for a command-line is as follows:

```
BE_HOME\be-engine [-h] [--propFile startup property file] [--propVar varName=value] [-p custom
property file] [-n engine name] [-d] -c CDD file -u processing unit ID   EAR file
```

For a detailed explanation of the above format see *Starting a BusinessEvents Engine at the Command Line* in *TIBCO BusinessEvents Administration*.

Congratulations!

You have completed the tutorial and now understand the basics of project design and deployment.



A follow-on example project, `FraudDetectionCache`, is available in the `BE_HOME/examples/standard` directory. It has a CDD configured for deploying a cache server and an inference agent.

You can explore Cache object management, the site topology editor, and use of the BusinessEvents Monitoring and Management component by reading the *TIBCO BusinessEvents Administration* guide.

Glossary

A

advisory event

A notice from BusinessEvents about activity in the engine, for example, an exception. See also [event](#), [simple event](#), [simple event definition](#), [time event](#).

agenda

A prioritized list of rule actions that may execute. Also known as the rule agenda. BusinessEvents recreates the agenda each time a change in the Rete network requires rules to be re-evaluated, for example, when a simple event is asserted. A rule action in an agenda may disappear without firing when the agenda is recreated, because conditions are no longer met.

agent

BusinessEvents operates at runtime using one or usually several agents of different types. See [cache agent](#), [inference agent](#), [query agent](#).

agent class

An agent type, defined in the cdd, that deploys as an agent instance.

assert

Put facts into the Rete network. When object instances or events are asserted into the Rete network, rules may fire as a result. See also [fact](#), [retract](#).

B

backing store

A disk-based database to provide persistence and a better level of reliability for cache objects.

C

cache

In BusinessEvents, a type of object management. Generally refers to distributed storage of facts (objects) in memory. See also [object management](#), [cache mode](#).

cache cluster

A group of processing units each running one or more agents configured for use by the cache-based object management system. Some agents are generally configured as cache agents. They store cache data for use by other cluster members. See [object management](#), [cache](#).

cache mode

Various cache mode options are available at the object level, for use with cache-based object management. They enable you to tune the performance of your application, and reduce its footprint in memory. See also [object management](#), [cache only](#), [cache plus memory](#), [memory only \(local only\)](#).

cache agent

A processing unit configured with a non-reasoning agent used as a storage node only.

Used with Cache object management. See also [agent](#), [cache](#).

cache only

One of the cache modes, used at the object level. Instances of entity types that use cache only are serialized and kept in the cache until needed. They must be explicitly loaded into the Rete network when needed for rule processing. See also [object management](#), [cache](#), [cache mode](#)

cache plus memory

One of the cache modes, used at the object level. Instances of entity types that use cache plus memory are serialized and kept in cache until needed. Used for objects that change infrequently. See also [object management](#), [cache](#), [cache mode](#).

cache server

See [cache agent](#).

caching scheme

Caching schemes define how various cache-based object management options are used. BusinessEvents provides preconfigured caching schemes that are used automatically based on configuration choices. The caching schemes are stored in a cache configuration descriptor file. See also [object management](#), [cache](#).

channel

A named configuration that allows BusinessEvents to listen to a stream of events from a given type of source, for example, JMS messages. A channel contains one or more destinations. See also [destination](#).

checkpoint

As used in persistence object management, a checkpoint is the point in time at which working

memory data is written to disk. The term checkpoint also encompasses all the activities involved in writing the data to disk.

cluster deployment descriptor

XML file containing properties to define the cluster, processing units, and agent classes. Configuration done using TRA file properties in earlier releases is now done using the CDD editor in BusinessEvents Studio.

complex event

An abstraction that results from patterns detected among simple events.

Example: A complex event might result from the following simple events that all occurred within one week's time: A stock broker buys shares of xyz stock. The same broker submits a very large order for xyz stock on behalf of a customer. The same broker sells shares of xyz stock at a profit. See also [simple event](#).

complex event processing (CEP)

Correlation of multiple events from an event cloud, with the aim of identifying meaningful events and taking appropriate action.

concept

An abstract entity similar to the object-oriented concept of a class. A concept is a description of a set of properties that, when grouped together, create a meaningful unit. Concepts can be organized in a hierarchical structure.

Example: *Department*, *employee*, *purchase order*, and *inventory item* are all concepts. The term *concept type* refers to the definition and the term *concept instance* refers to the actual object. Concepts are generally created using event data. See also [event](#).

concept reference

A property within one concept that references the ID of another concept, known as the

referenced concept. A type of relationship between concepts.

conflict resolution cycle

A cycle of activities during which the engine executes one set of rule actions on the currently asserted facts. One RTC may contain multiple conflict resolution cycles. See also [Run to completion \(RTC\) cycle](#), [agenda](#).

contained concept

A concept that exists entirely within another concept. A type of relationship between concepts.

custom function

You can add Java-based custom functions as needed to supplement the library of standard functions provided with BusinessEvents. See also [ontology function](#), [rule function](#), [standard function](#).

D

decision table

A tabular form presenting a set of conditions and their corresponding actions. A graphical tool for building rules. Used in TIBCO BusinessEvents Decision Manager.

deserializer

A class that performs conversion tasks. In BusinessEvents, a deserializer converts messages to events. See also [serializer](#)

destination

A channel property that defines a contact point on a given channel. For example, for a TIBCO Rendezvous channel, the destination properties would specify the subjects on which to listen.

distributed cache

In BusinessEvents, a form of cache-based object management. In a distributed cache, cached object data is partitioned between the processing units (JVMs) in the cache cluster for efficient use of memory. See also [object management](#), [cache](#).

E

entity

A concept, simple event, or scorecard. Entity types are the definition of the entity. Similar in meaning to object. The term "instance" generally refers to a concept instance.

event

An object representing some occurrence or point in time. See also [advisory event](#), [simple event](#), [simple event definition](#), [time event](#).

evict

To remove an object or entry from a cache. An eviction policy defines when an object is removed from the cache. See also [object management](#).

expires (event)

At the end of the event's time to live period, the event is said to expire. It is removed from the Rete network and (as needed) acknowledged. Other actions depend on the type of object management used. See also [time to live](#).

F

fact

An instance of an event or concept or scorecard in the Rete network.

I**in memory**

In BusinessEvents, a form of object management. Refers to storage of facts (objects) used by the runtime engine in JVM memory. See also [object management](#).

memory only (local only)

One of the cache modes, available for cache-based object management configuration. Instances of entity types that use this mode are not stored in cache or backing store and are available only in the processing unit's local JVM memory. See also [object management](#), [cache](#), [cache mode](#).

inference agent

In a deployed system, inference agents process incoming events using a Rete network and a set of rules that are triggered by conditions in incoming events. Inference agents in Cache OM systems allow fault tolerance and load balancing. See also [Rete algorithm](#).

instance

Similar to the Java term "object instance." By custom, applied only to concepts, though event definitions have object instances also.

L**lambda transition**

A transition without a condition. This term is used in state model configuration.

M**O****object management**

The aspect of BusinessEvents that deals with management of all the facts used in the runtime engine. See also [in memory](#), [cache](#), [persistence](#).

ontology function

BusinessEvents generates ontology functions for each entity type in a project. There are three types of ontology functions: *constructors*, to create a simple event or concept instance; *time events*, to create and schedule time events, and *rule functions*, to invoke rule functions. See also [custom function](#), [rule function](#), [standard function](#).

P**payload**

Similar to a JMS message, a simple event can contain properties and a payload. The payload holds the content of the message. You can define the XML schema for the payload when you configure the simple event definition. Payloads can also contain strings and Byte arrays.

persistence

In BusinessEvents, a form of object management. Refers to storage of facts (objects) used by the runtime engine, in a database. See also [object management](#).

processing unit

Definition of a BusinessEvents engine which runs in one JVM. Contains agents and other properties.

Q

query agent

A query agent is a non-reasoning agent that and has read-only access to the underlying objects in the cache cluster. A query agent has no Rete network. Available only with TIBCO BusinessEvents Event Stream Processing add-on software. See also [agent](#).

R

Rete algorithm

Dr Charles L. Forgy developed the Rete algorithm for expert systems. See also [Rete network](#).

Rete network

An in-memory network of objects based on the Rete algorithm which enables fast matching of facts with rule dependencies. "A Rete-based expert system builds a network of nodes, where each node (except the root) corresponds to a pattern occurring in the left-hand-side (the condition part) of a rule. The path from the root node to a leaf node defines a complete rule left-hand-side. Each node has a memory of facts which satisfy that pattern." (http://en.wikipedia.org/wiki/Rete_algorithm)

retract

Remove facts from the Rete network. See also [assert](#), [fact](#), [Rete network](#), [working memory](#).

RMS

Rules Management Server. The server component of TIBCO BusinessEvents Decision Manager add-on software. RMS manages the rules management repository.

Run to completion (RTC) cycle

A run to completion (RTC), cycle generally begins when an external action causes changes to the Rete network. It ends when there are no more rule actions to execute as a result of that initial change (and any subsequent changes caused by rule actions). One RTC is composed of one or more *conflict resolution cycles*. Other terms for RTC are *forward chaining* and *inferencing*. See also [conflict resolution cycle](#).

rule

A declaration, with a set of conditions and actions. If all the conditions in the rule are satisfied by facts in the Rete network (and the rule is at the top of the agenda), BusinessEvents executes the action. See also [working memory](#), [Rete network](#), [agenda](#).

rule based time event

See [time event](#).

rule function

Custom functions written using the BusinessEvents rule language and using a provided user interface. See also [custom function](#), [ontology function](#), [standard function](#).

Also used to refer to a type of ontology function.

rule session

An older term that has been replaced by the term inference agent. See also [inference agent](#).

S

serializer

A class that performs conversion tasks. In BusinessEvents, a serializer converts events to messages. See also [deserializer](#)

simple event

An object representing a business activity that happened at a single point in time. A simple event includes information for evaluation by rules, metadata that provides context, and a separate payload — a set of data relevant to the activity. See also [advisory event](#), [event](#), [simple event definition](#), [time event](#), [time to live](#), [expires \(event\)](#).

simple event definition

A description of the channel, destination, properties, and payload for a simple event. See also [simple event](#).

standard function

A library of standard functions is provided with BusinessEvents for use in rules and rule functions. See also [custom function](#), [ontology function](#), [rule function](#).

T**time event**

A type of event definition, used as a timer. Two types are available: rule based, and interval based. See also [advisory event](#), [event](#), [simple event](#).

time to live

A simple event property that defines the delay after which a simple event expires. See also [expires \(event\)](#).

U**UML (Unified Modeling Language)**

A language that assists in building a diagram of any complex entity. BusinessEvents diagrams

use the UML. The BusinessEvents term, *concept*, is similar to a UML class.

V**virtual rule function**

A rule function whose signature is defined in a BusinessEvents project and whose implementation is defined using decision tables. For use by TIBCO BusinessEvents Decision Manager add-on software. See also [rule function](#).

W**working memory**

The runtime processing area for rules, objects, and actions. Rules apply only to data in the working memory. Often used to mean Rete network. See also [rule session](#), [Rete network](#)

Index

A

account concept [17](#)
 advisory event (glossary) [45](#)
 agenda (glossary) [45](#)
 agent (glossary) [45](#)
 assertion (glossary) [45](#)

B

BE_HOME [ix](#)

C

cache (glossary) [45](#)
 cache cluster (glossary) [45](#)
 cache mode (glossary) [45](#)
 cache only (glossary) [46](#)
 cache plus memory (glossary) [46](#)
 cache server (glossary) [45, 46](#)
 caching scheme (glossary) [46](#)
 channel (glossary) [46](#)
 channels
 configuring (tutorial example) [10](#)
 checkpoint (glossary) [46](#)
 complex event (glossary) [46](#)
 concept (glossary) [46](#)
 concept reference (glossary) [46](#)
 conflict resolution cycle (glossary) [47](#)
 Console view [7](#)
 contained concept (glossary) [47](#)
 create a Rendezvous channel and a destination [10](#)
 customer support [xii](#)

D

debit event [13](#)
 Decision Manager (glossary) [47](#)
 decision table (glossary) [47](#)
 deserializer (glossary) [47](#)
 destination (glossary) [47](#)
 destinations
 configuring (tutorial example) [10](#)
 distributed cache (glossary) [47](#)

E

entity (glossary) [47](#)
 ENV_HOME [ix](#)
 Error Log view [7](#)
 event (glossary) [47](#)
 examples [2](#)

F

fraud detection runtime flow [6](#)
 fraudcriteria scorecard [20](#)
 frauddetection project [7](#)

I

in memory (glossary) [48](#)
 in memory only (glossary) [48](#)
 index (glossary) [50](#)
 inference agent (glossary) [48](#)
 initializeaccount rule function [22, 24](#)
 instance (glossary) [48](#)

L

lambda transition (glossary) [48](#)

O

object management (glossary) [48](#)

P

payload (glossary) [48](#)

persistence (glossary) [48](#)

Problems view [7](#)

Properties view [7](#)

Q

query agent (glossary) [49](#)

R

Rete network (glossary) [49](#)

retraction (glossary) [49](#)

RMS (glossary) [49](#)

rule (glossary) [49](#)

rule session (glossary) [49](#)

S

serializer (glossary) [49](#)

simple event (glossary) [50](#)

simple event definition (glossary) [50](#)

simple events

 configuring (tutorial example) [13](#)

support, contacting [xii](#)

T

technical support [xii](#)

TIBCO_HOME [ix](#)

U

UML (glossary) [50](#)

V

views

 Console [7](#)

 Error Log [7](#)

 Problems [7](#)

 Properties [7](#)

virtual rule function (glossary) [50](#)

W

Workbench (glossary) [50](#)

working memory (glossary) [50](#)