



TIBCO BusinessEvents®

Configuration Guide

*Software Release 5.6
August 2019*

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 THE LICENSE FILE) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE "LICENSE" FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

ANY SOFTWARE ITEM IDENTIFIED AS THIRD PARTY LIBRARY IS AVAILABLE UNDER SEPARATE SOFTWARE LICENSE TERMS AND IS NOT PART OF A TIBCO PRODUCT. AS SUCH, THESE SOFTWARE ITEMS ARE NOT COVERED BY THE TERMS OF YOUR AGREEMENT WITH TIBCO, INCLUDING ANY TERMS CONCERNING SUPPORT, MAINTENANCE, WARRANTIES, AND INDEMNITIES. DOWNLOAD AND USE OF THESE ITEMS IS SOLELY AT YOUR OWN DISCRETION AND SUBJECT TO THE LICENSE TERMS APPLICABLE TO THEM. BY PROCEEDING TO DOWNLOAD, INSTALL OR USE ANY OF THESE ITEMS, YOU ACKNOWLEDGE THE FOREGOING DISTINCTIONS BETWEEN THESE ITEMS AND TIBCO PRODUCTS.

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

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

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

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. Please see the readme.txt file for the availability of this software version on a specific operating system platform.

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

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

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

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

Copyright © 2004-2019. TIBCO Software Inc. All Rights Reserved.

Contents

Figures	8
TIBCO Documentation and Support Services	9
Cluster Deployment Descriptor (CDD)	11
CDD and Object Management (OM) Type	12
Global Variables Setup in CDD (for Command Line Startup)	12
CDD Configuration	13
Adding a CDD	13
CDD Cluster Tab General Settings Reference	14
After the CDD File is Added	15
Configuring Management of Domain (Entity) Object Instances	15
Configuring Limited Cache and Object Table Cache Options	15
CDD Cluster Tab and Cache OM Settings Reference	16
Synchronous and Asynchronous Replication of Cache Objects	19
Property for Cache Based Object Management on AIX	20
Cluster Discovery and Internal Communication	20
Configuring a DataGrid Cluster (Metaspace)	21
Configuring an Oracle Coherence Cluster	21
Configuring Cache Related OM Settings	22
DataGrid Discover URL	22
Multicast (PGM) Cluster Member Discovery	22
Unicast (Well-Known Address) Cluster Member Discovery	23
DataGrid Listen URL	24
Remote Client	25
Connecting an Inference Agent as a Remote Client to TIBCO DataGrid	26
DataGrid Transport Security	26
Setting Up DataGrid Security and Authentication	27
Working with an Example	28
Restricting Transport Access	29
Schema Model Migration with Shared Nothing Persistence	30
Enabling Hot Deployment of New Concept and New Concept Properties	30
CDD Cluster Tab DataGrid Properties Reference	30
Enabling Use of Oracle Coherence as the Cache Provider	37
Oracle Coherence Cluster Discovery	37
Guidelines for Managing Coherence Clusters	37
Configuring Multicast Cluster Discovery for Coherence Clusters	38
Configuring Well-Known Address Cluster Discovery	38

CDD Cluster Tab Coherence Properties Reference	39
CDD Load Balancer	44
Content-Aware Load Balancing	44
Content-Aware Load Balancer	45
Adhoc Load Balancer	46
Creating the Load Balancer	46
Configuring the Receiver	47
CDD Load Balancer Tab Properties Reference	47
CDD Backing Store	49
Configuring Backing Store Settings and Properties	49
CDD Cluster Tab Backing Store Settings Reference	49
CDD Cluster Tab Backing Store Properties Reference	52
Setting Up Shared Nothing Persistence	57
Runtime Configuration to Specify the Engine Name Property	58
Recovery Options for Shared Nothing Persistence	58
Berkeley DB Shared All Persistence	60
Configuring the Berkeley DB Shared All Persistence Option	60
Reference to Berkeley DB Shared All Persistence CDD Properties	61
Reference To Berkeley DB (JE) Properties	62
Domain Objects Configuration	63
Preloading Options	63
Configuring Preloading Options	64
CDD Cluster Tab Domain Objects Default Settings Reference	64
CDD Cluster Tab Domain Object Override Settings Reference	67
Collections, Agent Classes, and Processing Units	72
Configuring Collections of Rules, Rule Functions, and Destinations	73
Updating Collections	74
CDD Collections Tab Input Destination Settings Reference	74
Agent Classes (All OM Types)	75
Adding an Agent Class	76
Adding a LiveView Agent to the BusinessEvents Project	77
Adding Entity Filter Configurations for the LiveView Agent	78
CDD Agent Classes Tab Settings Reference	80
CDD Agent Classes Tab Properties Reference	82
Log Configurations	86
Configuring Log Configurations	88
Overriding The Default Logging Mode	88
Custom Log4j Configuration Examples	89
CDD Collections Tab Log Configurations Settings Reference	90

Logging for TIBCO BusinessEvents DataGrid	92
Configuring the Date Format in the Log Files	93
Processing Units (All OM Types)	94
Adding a Processing Unit	94
CDD Processing Units Tab Settings Reference	95
CDD Processing Units Tab Coherence Log Properties Reference	96
CDD Processing Units Tab JMS Server Connection Properties	98
StreamBase Channel Connection Properties	102
JDBC Backing Store	103
Backing Store Setup and Configuration	104
Ontology Identifiers that Exceed the DBMS Maximum Column Length	105
Ontology Identifiers that Use Database Key Words	106
String Properties that Exceed the DBMS Maximum Column Length	106
Resources Required for Setting Up the Database	106
Install Prerequisites for DBMS Software	108
SQL Server	108
Configuring Your Machine for Windows Authentication	109
SQL Server authentication vs. Windows authentication	110
Datatype and Driver Information	110
Configuring Your Machine for Oracle Database	110
Installing an OCI Driver - OCI Driver Support	111
Oracle Real Application Cluster (RAC)	111
Other Information about Oracle Database and Drivers	112
Configuring CDD for Special Cases (As Needed)	112
Adding a JDBC Connection Resource (Now or Later)	113
Configuring Backing Store Settings in the CDD (Now or Later)	114
Preventing Database Outages when a Cluster Ceases with Processing	114
Building the EAR File	116
Initializing the Database and Generate Non-Project Tables	116
Project-Schema-Specific SQL Scripts	118
Generating the Project-Schema-Specific SQL Scripts (with Wizard)	118
Generating Scripts Manually	120
Schema Definition Commands Options	120
Aliases File and Project Schema Script	121
Configuring Aliases File and Project Schema Script	121
Step 1 Check the Aliases File and Modify Aliases as Desired	121
Step 2 Run the Project Schema Script (as BE_USER)	122
Step 3 If Needed - Map Key (Reserved) Words to Aliases	122
Providing Project-Specific Keyword Aliases	123

Step 4 Project Configuration (As Needed)	123
Updating Existing Backing Store Schema	123
What the Schema Update Utility Can Handle Automatically	124
Backing Store Table Reference	125

Figures

Remote Client Architecture 25

CDD Based Router and Receiver Configuration 45

Entity Filter Configurations for the LiveView Agent 79

Log Configurations Properties 89

Main Tasks in Setting up a Backing Store **103**

Complete Task Flow **104**

TIBCO Documentation and Support Services

How to Access TIBCO Documentation

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

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

Product-Specific Documentation

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

`TIBCO_HOME/release_notes/TIB_businesses-events-enterprise_5.6.0_docinfo.html` where `TIBCO_HOME` is the top-level directory in which TIBCO products are installed. On Windows, the default `TIBCO_HOME` is `C:\tibco`. On UNIX systems, the default `TIBCO_HOME` is `/opt/tibco`.

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

- *TIBCO BusinessEvents Release Notes*
- *TIBCO BusinessEvents Installation*
- *TIBCO BusinessEvents Getting Started*
- *TIBCO BusinessEvents Architect's Guide*
- *TIBCO BusinessEvents Administration*
- *TIBCO BusinessEvents Developer's Guide*
- *TIBCO BusinessEvents Cloud Deployment Guide*
- *TIBCO BusinessEvents Data Modeling Developer's Guide*
- *TIBCO BusinessEvents Event Stream Processing Pattern Matcher Developer's Guide*
- *TIBCO BusinessEvents Event Stream Processing Query Developer's Guide*
- *TIBCO BusinessEvents Configuration Guide*
- *TIBCO BusinessEvents WebStudio User's Guide*
- *TIBCO BusinessEvents Decision Manager User's Guide*
- Online References:
 - *TIBCO BusinessEvents Java API Reference*
 - *TIBCO BusinessEvents Functions Reference*

How to Contact TIBCO Support

You can contact TIBCO Support in the following ways:

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

How to Join TIBCO Community

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

Cluster Deployment Descriptor (CDD)

Object management is configured using the Cluster Deployment Descriptor (CDD), an XML file that you edit in TIBCO BusinessEvents Studio using a provided editor.

One EAR file and one CDD file define all the settings for all the engines and agents you want to deploy for a single application.

Because the deploy-time configuration settings for all processing units are in the CDD file, you do not have to rebuild the EAR file to make changes to deploy-time settings. However, as desired, you can use only the copy of the CDD file that is in the EAR, for tighter control and uniformity.

When you deploy a processing unit, you specify these items:

- An EAR file
- A CDD file that you have configured for that EAR
- A processing unit (engine) that is configured in the specified CDD file.

The CDD file you specify can be in the file system or in the EAR file. To specify a CDD located in an EAR file, provide its project path and name.



For deployment using TIBCO Administrator, the CDD file in the EAR is used. The CDD file does not accept global variables as values.

You can configure multiple CDD files for a project for different purposes such as testing a design, trying out different object management options, dividing the work differently between agents and processing units (engines), and so on. However you use the same CDD file when deploying all the processing units for an application.

CDD Settings at Runtime

It is important to understand the effect of design-time settings in the runtime environment. The tab and section within a tab where you set values in the CDD can affect the scope of those values, and how they can be overridden.



Channel design time does not need to be changed to disable it in TIBCO BusinessEvents 5.1. You can specify on the agent class in the CDD using the `be.channel.deactivate` property and set its value to a comma, separated lists of channels. These channels will thus not be activated. If you want to reactivate a channel, remove it from this property.

Using Properties at Different Levels

The scope of a property depends on the property sheet you add it to. Not all properties are valid at all levels. Use your judgment.

For example, properties that include the agent class name, such as `Agent.AgentClassName.checkDuplicates`, can be used at different levels. Here is the scope of each level for these `AgentClassName` properties:

Cluster level

Applies to all `AgentClassName` agents in the cluster.

Processing unit level

Applies to any `AgentClassName` agent deployed in the specified processing unit.

Agent class level

Applies to any `AgentClassName` agent, used in any processing unit.

(Not all agent-level properties include the agent class in the property name.)

Only one value for a property is used when a processing unit is deployed.

Order of precedence at runtime can affect decisions made at design time. This is discussed and explained in the documentation for TIBCO Administrator.

CDD and Object Management (OM) Type

The OM type determines many of the configuration options available in the CDD editor.

For an introduction to this topic, see Object Management Types in *TIBCO BusinessEvents Architect's Guide*.

Specifying and Changing the Object Management Type

If you choose the OM type in the wizard, a template for that OM type is used. For example, if you choose Cache, then a cache agent with default values is created for you. If you choose the OM type in the editor (after you finish the wizard), you add the necessary settings yourself and defaults are not provided. You can switch back and forth between object types without losing any configuration values, but only until you save. For example, if you change from cache to memory and save, the cache agent and its configuration are lost. For all OM types, however, the General settings are configured in the same way, as documented in this section.

Names in the CDD must conform to the `NCName` datatype. See the following page for more details:

<http://www.w3.org/TR/REC-xml-names/#NT-NCName>

Some Japanese characters, such as half-width Katakana, have issues when they are used in XML names. See the following document for more details:

<http://www.w3.org/Submission/japanese-xml/>

Global Variables Setup in CDD (for Command Line Startup)

Global variables are added to a project and their value is set using the TIBCO BusinessEvents Studio Global Variables editor.

You can also set global variable values in the CDD file. This can sometimes be useful if you will start the engine at the command line. It is not done for other purposes

Global variables set in TIBCO Administrator or TIBCO BusinessEvents Monitoring and Management (BEMM) override those set in the CDD. If you will deploy using TIBCO Administrator or BEMM, set global variable values using the features provided in TIBCO Administrator or MM (whichever you will use), not in the CDD.

Add properties using this format:

```
tibco.clientVar.GVName = value
```

The `GVName` must exactly match the name set in the TIBCO BusinessEvents Studio Global Variables editor.

If global variables are defined in the TIBCO BusinessEvents project using groups, specify the group path using forward slashes. For example, if a variable `JMSuri` is located under a group called `URIs`, specify the variable as `tibco.clientVar.URIs/JMSuri`.

Add such properties at the appropriate level in the CDD, depending on the desired scope: cluster, processing unit, or agent class.

CDD Configuration

To configure object management for Cache OM, select the cache provider, configure a cache agent quorum, and number of backup copies.

If you have a backing store, you can also configure a limited cache and specify the cache size for each entity.

Use this chapter if you picked Cache OM in the second page of the New Cluster Configuration wizard. This chapter explains how to configure cluster discovery and internal communications, and how to configure cache-related object management settings.

In addition to the basic reference tables (as noted in the procedure below) the following sections have additional specific guidelines:

- [Configuring Management of Domain \(Entity\) Object Instances](#)
- [Cluster Discovery and Internal Communication](#)
- [Synchronous and Asynchronous Replication of Cache Objects](#)
- [Property for Cache Based Object Management on AIX](#)

Configuring the Backing Store

For backing store details, see [CDD Backing Store](#) for project configuration details and see [JDBC Backing Store](#) for details about setting up the backing store itself.

Configuring Cache OM Options

To configure cache OM options, first add a CDD file (or open the CDD file you added). Then select the Cluster tab > Object Management node on the left and on the right, configure settings as explained in [CDD Cluster Tab and Cache OM Settings Reference](#).

Configuring a Limited (or Unlimited) Cache

You can use the options available to configure a cache of a limited size, or use the described procedure to set options for an unlimited cache.

Also see Limited and Unlimited Cache Size in TIBCO BusinessEvents Architect's Guide.

Use of limited cache is supported only when a backing store is also used. The backing store retains entries in excess of the limit. Without use of a backing store data inconsistencies could result:

- Entries for an object in the object table (an internally used cache) and in the object cache itself could expire independently of each other.
- Domain object settings for limited cache apply at the object level. Related concepts could have different settings. For example, a container concept could have a limited cache setting and its container concept an unlimited cache setting. Each could be evicted at different times.






Adding a CDD

When you add a CDD, you select an object management type (known as the OM type).

Procedure

1. In TIBCO BusinessEvents Studio Explorer, right-click the folder where you want to store the CDD and select **New > Cluster Deployment Descriptor**. You see the New Cluster Configuration Wizard.
2. In the **File** name field, type a name for the CDD and click **Next**. You can change the name in the editor as desired.

-  When you are using TIBCO Administrator for deployment, by default, TIBCO Administrator looks for a processing unit named default and a CDD file named default.
3. At the Object Management Selection page, select an object management type for the deployment, then click **Finish**. You will see the CDD editor.
-  Check the Object Management node on the left to be sure you selected the correct OM type. Depending on the OM type you chose, you see one of the following:
- Object Management: [Cache]
 - Object Management: [In Memory]
- You can right-click the Object Management element and select **Change to OM Type**.
4. Select Cluster tab General on the left. On the right, specify the following:
- The cluster name and message encoding.
 - As desired, an author name, and any comment you wish to record. (Version and date are not editable.)
- See [CDD Cluster Tab General Settings Reference](#) for details.
-  For deployment with TIBCO Administrator, the message encoding specified in the CDD file General settings must match the TIBCO Administrator domain's message encoding.
5. Save the resource.

What to do next



Configure the CDD file settings according to your requirement under different tabs.

CDD Cluster Tab General Settings Reference

The CDD Cluster tab allows you to set up the general cluster properties.

For the related procedure, see [Adding a CDD](#).

CDD Cluster Tab General Settings

Property	Notes
Cluster Name	<p>Specifies the name of the cache cluster.</p> <p>If TIBCO BusinessEvents DataGrid is the cluster provider, any spaces in the name are converted to underbar characters internally.</p> <p> Do not use the name \$cluster. It is a reserved name. Initially set to the CDD name.</p>
Message Encoding	<p>The encoding used in Rendezvous messages exchanged between TIBCO applications.</p> <p> For deployment with TIBCO Administrator, the message encoding specified here must match the TIBCO Administrator domain's message encoding. Default is ISO8859-1 (which is also the default for TIBCO Administrator).</p>
Author	<p>The name of the author of this CDD, as desired.</p> <p>Initially set to the currently logged-on user name.</p>

Property	Notes
Comment	Any comments as desired. Comments persist across versions.
Version	View-only field to record the version of the CDD, for information only. You could, for example, check whether deployed CDDs are all using the same version.
Date	View-only field to record creation time of this version.

After the CDD File is Added

After you added a CDD file, a few configuration steps remain.

The next steps for each object management type are provided in the following sections:

- In Memory OM: (No cluster configuration.) [Collections Agent Classes and Processing Units](#).
- Cache OM: [CDD Configuration](#).

Database Concepts Configuration

If you use database concepts, available in the TIBCO BusinessEvents Data Modeling add-on product, see *TIBCO BusinessEvents Data Modeling Developer's Guide* for details about configuring the Database Concepts node in the CDD.

Dashboard Agent Configuration

If you use the TIBCO BusinessEvents Views add-on, see *TIBCO BusinessEvents Views Developer's Guide* for details about configuring dashboard agents.

Configuring Management of Domain (Entity) Object Instances

If you choose Cache OM, you must also configure how to manage the domain (that is, entity) object instances.

For example you can determine whether the instances are flushed from the Rete network after each RTC (as is generally recommended) or are kept in the cache.

If you set up a backing store, you can specify additional settings. For example you can define a subset of ontology object instances to be stored in the backing store, and you can control which (if any) object instances are loaded into the cache from the backing store at system startup.

Configuring Limited Cache and Object Table Cache Options

As desired, you can set the cache to limited at the default level and unlimited for specified objects; or you can set the cache to unlimited at the default level and limited for specified objects.

When the cache is limited, the number of cache objects is `limit * no.of cache servers`.

Procedure

1. Select the CDD editor Cluster tab Object Management node.
2. In the Entity Cache Size setting, enter the desired number of objects per entity type.
3. In the Object Table Cache Size, enter the desired number of objects (handles) in the object table cache. You cannot set this value differently for different object types.
See The Role of the Object Table in *TIBCO BusinessEvents Architect's Guide* for more details about the object table.
4. Select the Cluster tab > Domain Objects > Defaults node.

Check the Is Cache Limited checkbox to enable limited cache globally. (Or uncheck the checkbox to use an unlimited cache globally.)

5. Select the Domain Objects > Overrides node. Select an override entry (or add one as needed).
6. Set the Is Cache Limited checkbox for the selected object type in one of the following ways:
 - If limited cache is set at the default level, uncheck the overrides Is Cache Limited checkbox to use an unlimited cache for objects of this type.
 - If unlimited cache is set at the default level, check the Is Cache Limited checkbox to use a limited cache for objects of this type.

See [Configuring a Limited \(or Unlimited\) Cache](#) for important requirements. For reference documentation on the Is Cache Limited checkbox see [CDD Cluster Tab Domain Object Default Settings](#).

7. Ensure that multiple clusters do not conflict.
 With TIBCO BusinessEvents DataGrid clusters, use a different value for Cluster Name (in the Cluster tab, General node) and also use different discovery values.

With Coherence clusters, use a different value for cluster name.

CDD Cluster Tab and Cache OM Settings Reference

The CDD Cluster tab is used for setting up the cache OM.

For the related procedure, see [Cluster Discovery and Internal Communication](#).

CDD Cluster Tab Cache OM Settings

Property	Global Variables	Notes
Provider	No	<p>Select the cache provider:</p> <p>TIBCO: Uses the provided TIBCO BusinessEvents DataGrid component. See DataGrid Discover URL and DataGrid Listen URL for more details.</p> <p>ORACLE: Uses a supported version of Oracle Coherence, for which you have a license that is appropriate for your usage. Requires additional configuration. See Enabling Use of Oracle Coherence as the Cache Provider and Oracle Coherence Cluster Discovery for details.</p>
Cache Agent Quorum	Yes	<p>Specifies a minimum number (quorum) of storage-enabled nodes that must be active in the cluster when the system starts up before the following occur:</p> <ul style="list-style-type: none"> • Data is preloaded from the backing store, if a backing store is configured and preloading is configured . • The other agents in the cluster become fully active. <p>The property does not affect the running of the deployed application after startup (though a message is written to the log file if the number of cache agents running falls below the number specified in this property). As a guideline, set to the number of cache agents configured.</p> <p>Default is 1.</p>

Property	Global Variables	Notes
Number of Backup Copies	Yes	<p>The number of backup copies (also known as the backup count) specifies the number of members of the distributed cache service that hold the backup data for each unit of storage in the cache. Recommended values are 0, 1, or 2.</p> <p>Value of 0 means that in the case of abnormal termination, some portion of the data in the cache will be lost. Value of N means that if up to N cluster nodes terminate at once, the cache data will be preserved. A backup count of 1 means one server plus one backup are needed, that is, two cache agents (or storage enabled nodes if cache agents are not used).</p> <p>To maintain the partitioned cache of size M, the total memory usage in the cluster does not depend on the number of cluster nodes and will be in the order of $M*(N+1)$.</p> <p>See Synchronous and Asynchronous Replication of Cache Objects for details on replication behavior and options for TIBCO BusinessEvents DataGrid.</p> <p>Default is 1.</p>
Entity Cache Size	Yes	<p>Specifies the size of the limited cache, in number of cache entries for each object type. The setting is per processing unit. See Configuring a Limited (or Unlimited) Cache.</p> <p>Default is 10000 (entries per object type)</p>
Object Table Cache Size	Yes	<p>Specifies the maximum size of the object table cache, in number of entries. Used with limited cache only.</p> <p>See The Role of the Object Table in TIBCO BusinessEvents Architect's Guide for more details about the object table.</p> <p>Also see Configuring a Limited (or Unlimited) Cache.</p> <p>Default is 100000 entries</p>
TIBCO BusinessEvents DataGrid		
Discovery URL	Yes	<p>Specifies how an engine (node) listens for discovery requests from nodes attempting to join the cluster. PGM protocol is supposed for multicast discovery. TCP Protocol is supported for unicast discovery</p>
ListenUrl	Yes	<p>The discovery mechanism is based on pure TCP. All the designated well-known metaspace members are identified by an IP address and a port number.</p>
Remote Listen URL	Yes	<p>Specifies on which IP address and TCP port this proxy metaspace member will be listening for the remote client connections.</p>

Property	Global Variables	Notes
Protocol Timeout	Yes	Indicates the protocol timeout value for space. The protocol can be unicast or multicast. Default value is -1 (forever).
Read Timeout	Yes	Indicates the read timeout value for the space, if a read timeout has been set. It specifies the read timeout value for a specified SpaceDef. The read timeout value applies to Get operations. Default value is 60000 (ms).
Write Timeout	Yes	Indicates the write timeout value for the space, if a write timeout has been set. Specifies the write timeout value that is set for the space. The write timeout value applies to Put, Take, Lock, and Unlock operations. Default value is 60000 (ms).
Lock Timeout	Yes	For a space that is locked, it specifies how long a member process will wait for it to become unlocked. The default is -1 (forever). Other valid values are 0 or any positive value.
Shutdown wait	Yes	Indicates the shutdown wait value for the space. Default value is 8500 (ms).
Worker Thread Count	Yes	Indicates the thread count specified for the space. Default value is 4.
Store Properties as Individual Fields	No	Enabling this property sets the property <code>be.engine.cluster.as.tuple.explicit</code> to true.
Security Enabled	No	Enables Transport level security for the DataGrid when selected. The following fields are displayed only if the Security Enabled checkbox is selected: <ul style="list-style-type: none"> • Policy File • Policy File Identity Password • Token File • Token File Identity Password • LDAP Identity File • Domain • Username • Password

Property	Global Variables	Notes
Policy File	Yes	Absolute path to the policy file which contains the security settings that the controller node enforces. It is generated using the <code>as-admin</code> utility.
Policy File Identity Password	Yes	The password for the identity key in the security policy file.
Token File	Yes	Absolute path to the token file which is used by requestor to connect to a metaspace whose security is defined in the policy file.
Token File Identity Password	Yes	The password for the identity key in the security token file.
LDAP Identity File	Yes	The absolute path for a file containing the key to use for LDAP with the certificate-based authentication.
Domain	Yes	Optional. Domain name for system based user authentication.
Username	Yes	User name for LDAP and system based authentication.
Password	Yes	Password for LDAP and system based authentication. In case authentication type in the policy file is "x509" then this is the password is for the private key in the LDAP Identity File .

Synchronous and Asynchronous Replication of Cache Objects

Backup count defines the number of backup object copies to make in addition to the primary cache object.

Backup cache writes can be done synchronously or asynchronously. There is a difference between the behavior of TIBCO BusinessEvents DataGrid and for Oracle Coherence as regards replication.

TIBCO BusinessEvents DataGrid Uses Asynchronous Replication

TIBCO BusinessEvents DataGrid is set up to use asynchronous replication. There is no option to use synchronous replication. Asynchronous replication allows you to run tests using a single cache agent.

With asynchronous replication, the inference agent writes to a cache agent and returns. The cache provider then makes a separate call to another cache agent to make the replica. This means that the writes from the inference agent do not incur the cost of synchronous replication, because replication happens on a different thread in the background. However, a small window exists in which the inference agent has written to the cache, and the cache provider has not replicated the data yet. If the cache agent fails at this point, data is lost because there is no replica. To safeguard the data, use a backing store with cache-aside database write strategy.

Oracle Coherence Uses Synchronous Replication

With Oracle Coherence as the cache provider, TIBCO BusinessEvents is set up to use synchronous replication. There is no option to use asynchronous replication.

With synchronous replication, when the inference agent writes to the cache, the cache provider makes a network call to another cache agent and makes a replica (or replicas) and then the call from the inference agent returns. This means making two cache puts for each cache operation. Therefore, synchronous replication is slower than asynchronous replication.

You must also ensure that the required number of cache agents is always up and running (depending on the backup count).

Property for Cache Based Object Management on AIX

When TIBCO BusinessEvents is installed on AIX and uses cache-based object management, you must add the property `java.net.preferIPv4Stack=true` to all TRA files.

If you do not add this property, you see the following exception:

```
java.net.SocketException: The socket name is not available on this system
```



Remember to set this property on all internal TIBCO BusinessEvents engines' TRA files too, such as in `be-mm.tra` for the TIBCO BusinessEvents Monitoring and Management (MM) server and the MM broker properties set in the MM CDD file. Add-on products also have engine TRA files you must update.

Cluster Discovery and Internal Communication

When you add a CDD file and select Cache OM type, you must configure how the members of the cache cluster discover each other at runtime and communicate with each other once the cluster is established.

This section has summary steps for both cache providers, and pointers to sections with more details.

Support for Host-Aware Replication

Host-aware replication for ActiveSpaces is controlled by the property `be.engine.cluster.as.hostaware.enable`. Host-aware replication requires that the member name be a 2-part name separated by a ".".

By default, this property is `true` (or enabled).

- If `true`, the ActiveSpaces member name will be set as:

```
hostname.be-engine-name
```

where *be-engine-name* is what is given on the `-n` command line option. When host-aware replication is enabled, if the cache nodes are not deployed on multiple machines to satisfy replication by the host, then replication will not happen (or will happen only according to the number of hosts available).

For example, if "Number of Backup Copies" is set to "1" and all cache nodes are deployed on a single host, then replication will be disabled (regardless of the number of cache nodes on that single host). If "Number of Backup Copies" is set to "2", and cache nodes are deployed only on 2 hosts, then only "1" backup copies will be maintained.

- If the property is `false`, host-aware replication will be disabled and the ActiveSpaces member name will be set as *be-engine-name*. Disabling hostaware replication will honor "Number of Backup Copies", provided that there are enough cache nodes deployed in the cluster.

Configuring a DataGrid Cluster (Metaspace)

An active LAN connection (device enabled and network cable plugged in) is required for TIBCO BusinessEvents DataGrid to work.

Procedure

1. Add a CDD file or open the CDD file you added.
2. Select the Cluster tab > Properties node on the left and on the right, add the following two properties, as needed:

```
be.engine.cluster.as.discover.url
be.engine.cluster.as.listen.url
```

The properties can be omitted if you use PGM multicast with default values. See the following sections for details on configuring these properties:

- [DataGrid Discover URL](#)
 - [Unicast \(Well-Known Address\) Cluster Member Discovery](#)
 - [DataGrid Listen URL](#)
3. If you use unicast (well-known address) discovery, and you use TIBCO BusinessEvents Monitoring and Management for monitoring and management, you must also do the following (in the to-be-monitored project CDD):
 - a) Add the following property to the cluster properties sheet:


```
be.mm.cluster.as.listen.url MMHostIP: Port
```

Specify the IP of the computer hosting the MM server, and an unused port.
 - b) Add the value of the `be.mm.cluster.as.listen.url` property to the list of addresses in the `be.engine.cluster.as.discover.url` property, which should be present at the cluster level (so the value is identical for all potential cluster members).

The discover URL for well-known address configuration uses the following format:

```
tcp://ip:port[;ip:port]*
```

Configuring an Oracle Coherence Cluster

First part of the task is to add /open a CDD file and set up the properties.

Procedure

1. Before you begin ensure that you have met all prerequisite steps.
See [Enabling Use of Oracle Coherence as the Cache Provider](#)
2. Add a CDD file or open the CDD file you added.
3. Select the Cluster tab > Properties node on the left and on the right, add properties as explained in [Oracle Coherence Cluster Discovery](#).
4. Add any other cluster level properties as needed.
See [Other Coherence Properties](#).

Configuring Cache Related OM Settings

Second part of the task is to configure the settings.

Procedure

1. Open the CDD file you added.
2. Select the Cluster tab > Object Management node and on the right configure settings as explained in [CDD Cluster Tab and Cache OM Settings Reference](#).
Also see [Synchronous and Asynchronous Replication of Cache Objects](#).

DataGrid Discover URL

When a cluster starts up, and also when new members join a cluster, a discovery process enables the members to discover each other.

The discover URL specifies how an engine (node) listens for discovery requests from nodes attempting to join the cluster.

After the discovery is complete, the members communicate internally using a listen URL (explained in [DataGrid Listen URL](#)).

Two types of discovery are available:

- Multicast discovery (PGM)
- Unicast discovery (TCP), also known as "well-known address" discovery

Configuration for both discovery methods is explained below.



A TIBCO BusinessEvents DataGrid cluster is also known as a *metaspace*.

A TIBCO BusinessEvents engine is a *node* in the metaspace.

If No Other Cluster Members are Started

If a newly started node does not discover any running cluster nodes, the behavior is different depending on the type of discovery used:

- If multicast discovery is used, the newly started node becomes the first node of a newly started cluster.
- If unicast (well-known-address) discovery is used there are two cases:
 - If the address of the newly started node is not in the discover URL's list then it continues to wait for other well-known nodes to start, and a warning is written to the console while it waits.
 - If the address of the newly started node is in the discover URL's list, then it becomes the first node of a newly started cluster.


Multicast (PGM) Cluster Member Discovery

The discover URL for multicast discovery uses PGM (Pragmatic General Multicast) protocol.

The discovery property is `be.engine.cluster.as.discover.url`. For multicast discovery, the value is a URL with the following format: `tibpgm://destinationPort/network/`

The default values equate to the following: `//7888/;239.8.8.9/`

Specify the parameters as follows.

Parameter	Notes
<i>destinationPort</i>	<p>Specifies the destination port used by the PGM transport.</p> <p>Must be the same value on all machines in the cluster.</p> <p>Default value is 7888.</p>
<i>network</i>	<p>Specifies the IP address of the interface to be used for sending multicast packets, and the multicast group address to be used.</p> <p>The format is as follows: <i>interface; multicast group address</i></p> <p>The value for <i>interface</i> is unique to a node. It must also be the same in both the discovery and the listen URLs for a node. If there are multiple interfaces on one machine, specify the interface you want to use and do not rely on the default value.</p> <p>The value for <i>multicast group address</i> must be the same on all machines in the cluster.</p> <p>The default value for <i>interface</i> is the first available interface provided by the operating system hosts file for the machine.</p> <div>  <p>If the desired interface is not listed in the hosts file then PGM picks the first available interface in the file. (On most operating systems, this file is called the <code>/etc/hosts</code> file.) If the first interface is the loopback interface (127.0.0.1) then PGM fails to start. In this case you would see a stacktrace exception in the log file such as the following:</p> <pre>SYS_ERROR (multicast_error - (8) grp_iface not a valid multicast interface)</pre> <p>To resolve this issue, either modify the hosts file, or provide the desired interface explicitly in the <i>network</i> argument.</p> <p>The default value for <i>multicast group address</i> is the multicast group address 239.8.8.9.</p> </div>

Unicast (Well-Known Address) Cluster Member Discovery

If you cannot or do not wish to use multicast discovery in your environment, then configure unicast discovery, also known as "well-known address" or WKA discovery.

These "well-known addresses" enable a newly started node to discover existing members. Unicast discovery uses the TCP protocol.

The discovery property is `be.engine.cluster.as.discover.url`. For unicast discovery, the value is a semicolon-separated list comprising a sub-set of all the listen URLs (which are different for each PU), using this format:

```
tcp://ip:port[;ip:port]*/
```

One cluster node in the WKA list must be running at all times

At least one cluster node specified in the well-known address list must be running at all times, so that other new members can join the cluster (metaspace). If all nodes specified in the well-known address list stop, then other nodes that are still running continue to function, but they print warnings to the console and no new members can connect to this cluster.



For WKA discovery, make discover URL a cluster-level property and listen URL a PU-level property

The discover URL property (`be.engine.cluster.as.discover.url`) must be present and configured identically for all potential cluster members. Therefore add this property at the cluster level of the CDD file. The listen URL property (`be.engine.cluster.as.listen.url`) must be present and configured differently for each possible cluster member. Therefore add this property at the PU level.

DataGrid Listen URL

The listen URL is used for direct communication between the members of the metaspace. It is configured the same way for multicast and for unicast discovery.

The listen URL value must be different for each cluster member, so configure it at the PU level.

The listen URL uses this format: `tcp://interface:port[-EndPort |*]/`

The cluster member binds to the specified interface and the specified port when creating the TCP socket. Specify the parameters as follows.

Parameter	Notes
<i>interface</i>	<p>To specify a value, use the desired IP address.</p> <p>The value for <i>interface</i> must be the same in both the discovery and the listen URLs for a node. If there are multiple interfaces on one machine, specify the interface you want to use and do not rely on the default value.</p> <p>The default value for <i>interface</i> is the first available interface provided by the operating system for the machine.</p>
<i>port</i>	<p>To specify a single port use the port number in the listen URL, as shown in this example:</p> <pre>tcp://interface:6000/</pre> <p>You can use an auto-incrementing feature, as explained in Auto-incrementing Within a Range of Ports.</p> <p>The default value is the first available port in the 50000+ range.</p>

Multiple Nodes on One Machine

If multiple nodes (engines) are running on one machine, identify each uniquely. Use the same value for *interface*, but a different value for *port* for each node.

Auto-incrementing Within a Range of Ports

If a machine has blocked some ports in the default range, or if you want to use a different range, you can configure the listen URL to start with a specified IP address and port, and optionally provide an upper limit. If the specified port is not available, TIBCO BusinessEvents auto-increments the port until

it finds an available port, up to the specified upper limit, if any. To specify a specific range use this format:

```
tcp://interface:port-EndPort/
```

For example, given the following listen URL, TIBCO BusinessEvents attempts to open port 8000 and if it is not available it tries the next port number, until it finds an available port, up to 9000 (inclusive). If none is available, it keeps retrying. Make some ports in the specified range available so that the cluster nodes can start.

```
tcp://interface:8000-9000/
```

To specify a range with the upper limit of unsigned short minus one, use this format:

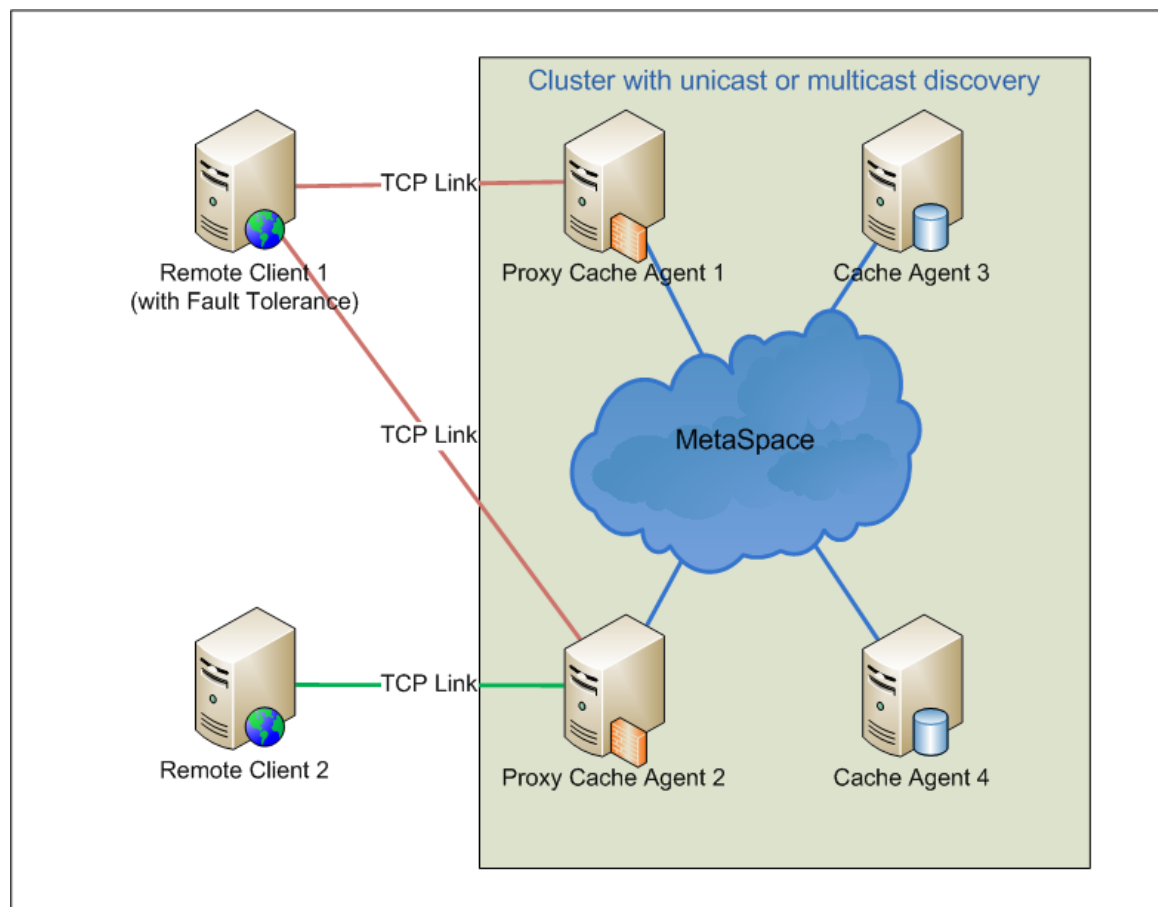
```
tcp://interface:port-*/
```

Remote Client

A remote client acts as a node without actually being a member of the cluster. Instead of being directly connected to space, it is connected through a proxy - typically through a cache-agent.

A remote client does not contribute any of its resources towards maintaining the cluster. TIBCO BusinessEvents extends the same feature to allow its non-cache agents to connect to the cluster as remote members through a cache agent, that acts as a proxy. Using the remote client you can do better data management, as remote clients do not take part in cluster and thus dropping of one or more nodes from the cluster do not affect its processing.

Remote Client Architecture



Remote Client Behavior

- Only the non-cache (like inference) nodes can be remote clients.

- A remote client cannot contribute to cache storage. Thus, the **Enable Cache Storage** checkbox under the **Processing Unit** tab in CDD is ignored.
- Remote clients only operate as long as the cluster is up and reachable. A remote client cannot resume operations across cluster restart.

Best Practices

- For fault-tolerance of proxy nodes, open a remote listen port on two or more of your cache agents and specify a list of these nodes in remote discovery URLs. Thus, the chance of remote client losing connections to the cluster is minimized.
- To ensure consistency in deployment, either configure all inference nodes as remote client or none of them as remote. A mix of remote and non-remote connections is not recommended.

Connecting an Inference Agent as a Remote Client to TIBCO DataGrid

You can configure one or more cache agent instances which play the role of the seeder and a proxy server for remote clients. On each of the cache node instances that act as a proxy for the remote clients, a remote listen URL (for the remote clients or Inference agents to connect) is configured. In the remote client, specify the list of IP addresses and ports of remote proxy cache agents to connect to.

Procedure

1. In TIBCO BusinessEvents Studio, open CDD of the cache agent, which is part of the cluster.
2. In the **Cluster** tab, specify the IP address and TCP port number to open remote proxy in the **Remote Listen URL** field.

Selecting IP address and port number is similar to specifying *interface* and *port* number for the DataGrid Listen URL.



In case, multiple cache agents are running on the same machine, override the remote listen URL with a different port for all such agents. You can override the port either by using the `be-engine` command line or the TIBCO BusinessEvents Enterprise Administrator Agent UI.

```
be.engine.cluster.as.remote.listen.url=tcp://<ip>:<port>
```

3. Save the project and restart the agent.
4. In TIBCO BusinessEvents Studio, open CDD of the remote inference agent for editing.
5. In the **Processing Units** tab, add the `be.engine.cluster.as.discover.url` property for the processing unit configured as remote client. Override this property with URL of the proxy cache agent to connect as a remote client.

```
be.engine.cluster.as.discover.url=tcp://ip:port[;ip:port]?remote=true
```



For fault tolerance, make two or more of your cache agents as remote proxies. To do so, open a remote listen port on two or more of your cache agents. Specify a list of these nodes in the remote discovery URLs of the remote client inference agents. This minimizes the chance of remote client losing connections to the cluster in case one of the proxy agent stops working.

You can specify the time for which the cluster waits for remote clients to reconnect after it is disconnected using the property `be.engine.cluster.as.remote.member.timeout`.

6. Save the project and restart the agent.

DataGrid Transport Security

Transport-level security allows you to protect data being transported within the DataGrid by preventing alteration of traffic, eavesdropping, and exchange of data between untrusted parties.

The available settings for `transport_security` are:

encrypted_normal

Use secure transport with 128 bit symmetric key encryption (default).

encrypted_strong

Use secure transport with 256 bit symmetric key encryption.

integrity

Use secure transport without encryption.

The two possible node types in a secure DataGrid are:

Controllers

Nodes dedicated to enforcing a security domain's defined security behavior for a cluster associated with the security domain. Security domain controllers are the only discovery nodes in a cluster.

Requestors

Nodes that require access to the data in the DataGrid, such as a seeder or a leech, and which need to be authorized by a controller. Requestors can never be used as discovery nodes.

Authentication

The controller nodes or processing units are configured with a security policy file. The requester nodes or processing units provide a token file and additional credentials to the controller for authentication. The controller performs authentication as defined in its policy file and using the credentials provided by the requester.

Setting Up DataGrid Security and Authentication

When security is used, any transmission of messages within the DataGrid occurs on a secure transport. A security domain's `transport_security` setting controls the level of security used for communication within the DataGrid.

Procedure

1. Start BusinessEvents with a non-secure DataGrid cluster.
In order to enable security for the BusinessEvents DataGrid, you first have to configure the cluster to use TCP based discovery (cannot use multicast discovery).
2. Create a security policy file using the ActiveSpaces admin utility (`AS_HOME/bin`).

```
as-admin> create security_policy
policy_name "mypolicy/mydomain"
policy_file "mypolicy.txt"
```
3. Edit the Metaspace Access List for the security domain in the security policy file.
 - Set the metaspace name (no quotes)
 - Set the discovery URL of the metaspace which can be TCP discovery only (no quotes)
4. Ensure that there is a `metaspace_access` entry with the cluster name from the BusinessEvents CDD.
The cluster name and the discovery URL should match the `metaspace=` and the `discovery=` values in the policy file.
5. Review the Transport Security settings to ensure they are set to meet your security requirements.
6. Save your changes to the security policy file.
7. Validate your security policy file using the ActiveSpaces admin utility.

```
as-admin> validate policy_name "mypolicy" policy_file "mypolicy.txt"
```

8. Create a security token file using ActiveSpaces admin utility.

```
as-admin> create security_token
domain_name "mydomain"
policy_file "mypolicy.txt"
token_file "mytoken.txt"
```

9. Ensure that there is a `metaspace_access` entry with the cluster name from the BusinessEvents CDD.

10. Validate your security token file using the ActiveSpaces admin utility:

```
as-admin> validate token_file "mytoken.txt"
```

11. Shut down the cluster.

12. Open the project CDD file for editing.

13. Each processing unit (PU) is either a Controller or a Requester. You can change its role in the **Processing Unit** tab using the `be.engine.cluster.as.security.mode.role` property to Requestor or Controller. settings. By default, TIBCO BusinessEvents assumes all nodes to be requesters. However, every cluster must have at least one controller node.

By default, all PUs are requesters so at least one PU in the cluster needs to be a controller. You can override the cluster level controller or requester settings in the PU by checking the **Override** checkbox and specifying a value. In most cases you would only need to override the key file paths .

14. In the Cluster tab, select the **Security Enabled** checkbox, for the Object Management, to enable the security.

15. Based on the role of PU as Controller or Requestor update their security settings: Supply the security file based on the se with the following property in the CDD:

- For Controller, specify the path of security policy file and password for its key in the **Policy File** and **Policy File Identity Password** fields.
- For Requestor, specify the path of security token file and password for its key in the **Token File** and **Token File Identity Password** fields.

Requester settings are dependent on the authentication policy defined in the controller's policy file.

- If the authentication type in the policy file is "userpwd" and authentication source is "system" or "ldap", specify **Username** and **Password**. You might also need to specify **Domain**, if the authentication source is "system".
- If the authentication type in the policy file is "x509", which means that the authentication source is an LDAP configured with certificate-based authentication, then specify **LDAP Identity File** and **Password** (in this case the password is for the private key in the identity file).

Working with an Example

The example for setting up policy and token files shows two nodes in a cluster called `mycluster` . One node is an inference engine and the other is a cache engine.

Procedure

1. Assume that the ListenURL for the cache (controller) is `tcp://203.0.113.0:9091` and the ListenURL for the inference (requester) is `tcp://203.0.113.0:9090`.
2. Generate the policy and token files from `as_admin`.
Edit the `metaspace_access` line in both files to set the correct cluster name.
3. Also edit the line to place the ListenURL of the controller into the `discovery=` setting.

```
metaspace_access=metaspace=mycluster;discovery=tcp://203.0.113.0:9091;
```

4. For the inference engine, you can then set the following properties:

```
be.engine.cluster.as.security.enable=true
be.engine.cluster.as.security.mode.role=Requester
be.engine.cluster.as.security.file=C:/temp/mytoken.txt
```

5. For the cache engine, you can then set:

```
be.engine.cluster.as.security.enable=true
be.engine.cluster.as.security.mode.role=Controller
be.engine.cluster.as.security.file=C:/temp/mypolicy.txt
```



There is no rule that caches have to be controllers, or inferences have to be requesters. The roles are interchangeable as long as there is one controller in the cluster.

Restricting Transport Access

Transport level security allows you to restrict transport connections within a security domain to only the “trusted” nodes.

Procedure

1. Open the security policy file for the domain in a text editor
2. Go to the line that reads

```
transport_access=false;cert_file=
```
3. Edit the line to read:

```
transport_access=true;cert_file=<trusted_certs_file>
```

where `trusted_certs_file` is the filename for a trusted certificate file that you will create in step 8.
4. Save the security policy file.
5. Use the **validate policy_file** command to validate the security policy file.
6. Use the ActiveSpaces admin utility to generate a security token file from the security policy file, which contains its own private key and a public certificate. This key and certificate are used to verify the identity of a node using the security token file when it tries to initiate any transport connections. For example:

```
as-admin> create security_token
domain_name "mydomain"
policy_file "mypolicy.txt"
create_identity
token_file "mytoken.txt"
```

7. Use the **validate token_file** command to validate the security token file.
8. Create an empty trusted certificates file to hold the public certificates of the nodes to allow transport connections from.
9. Copy and paste the public certificate of the local token identity from the security token file into the trusted certificates file.

The public certificate is everything in the security token file between and including

```
-----BEGIN CERTIFICATE-----
-----END CERTIFICATE-----
```

10. Save the trusted certificates file.
11. Start a security domain controller using the security policy file name when connecting to the DataGrid.
 DataGrid communication within the security domain is now restricted to only security domain controllers and security domain requestors that connect to the cluster using a security token file whose public certificate is contained in the trusted certificates file.

Schema Model Migration with Shared Nothing Persistence

Using Shared Nothing persistence to migrate a schema model.

You can choose one of these options:

- Deploy the new EAR with the additional field and restart all nodes. System will automatically alter the space as needed during recovery.
- If it is not desired to restart cache nodes, shut down all inference engines. Connect with AS administrator and alter the space to add new fields as shown in example below.

The new field must be nullable as is the case for all user fields in BusinessEvents .

```
alter space name "dist-unlimited-bs-readOnly-Test--be_gen_Concepts_Simple" add
(field name "long_field" type 'LONG' nullable true)
alter space name "dist-unlimited-bs-readOnly-Test--be_gen_Concepts_Simple" add
(field name "con_cept_array" type 'BLOB' nullable true)
```

Then start the inference engines.

- Hot deployment of new properties for Shared Nothing persistence has been added .



Adding a concept property of type contained concept with the contained concept type set to an existing concept is supported only for the options above.

Hot Deployment of New Properties

Hot deployment of new properties into existing TIBCO BusinessEvents concepts is available only in these cases:

- When cache Object Management is enabled with no persistence or *Shared Nothing* persistence.
- When the concepts with new properties are cache-only.

Enabling Hot Deployment of New Concept and New Concept Properties

New concept and new concept properties can be enabled for hot deployment for Cache OM with no backing store, while it is enabled by default for the shared nothing persistence.

Procedure

- In CDD file, under the **Cluster** tab select the **Store Properties As Individual Fields** check box for the **Object Management: [Cache]** configuration.

This property is selected by default when using shared nothing persistence. Also, adding a concept property of type contained concept with the contained concept type set to an existing concept is not supported.

CDD Cluster Tab DataGrid Properties Reference

DataGrid properties can be configured using the CDD Cluster Tab. Refer to the following table for a list of properties that you can use to retrieve data from the DataGrid.





Discovery and listen URL interfaces must match. Ensure that a node's interface (IP address) is specified using the same value in the discover URL and in the listen URL. If there are multiple interfaces on one machine specify the IP explicitly in both properties.

CDD Cluster Tab


Property	Notes
<code>be.engine.cluster.as.aggregate.prefetch.size</code>	<p>When queries are executed against DataGrid via the aggregate query functions (found under <code>Query.Datagrid.Aggregate</code>), you can define the number of entries to prefetch for optimum performance.</p> <p>Although TIBCO BusinessEvents has a default value that provides best performance for general use cases, you can adjust the value to meet your specific use case needs.</p> <p>Valid values are any positive long numbers or -1 (-1 = prefetch all).</p> <p>Default value is -1.</p>
<code>be.engine.cluster.as.browser.prefetch.size</code>	<p>When queries are executed against DataGrid via any 'select' type of queries, you can define the number of entries to prefetch for optimum performance.</p> <p>Although TIBCO BusinessEvents has a default value that provides best performance for general use cases, you can adjust the value to meet your specific use case needs.</p> <p>Valid values are any positive long number or -1 (-1 = prefetch all).</p> <p>Default value is -1.</p>
<code>be.engine.cluster.as.connection.retry.count</code>	<p>You can specify the number of attempts allowed for the metaspace connection when the discovery node is not available.</p> <p>The default value is 5.</p>
<code>be.engine.cluster.as.lookup.prefetch.size</code>	<p>When queries are run against DataGrid by using <code>getByExtByUri()</code> or by <code>loadByExtIdByUri()</code> functions (which return either 1 or no results), you can define the number of entries to prefetch for optimum performance.</p> <p>Although TIBCO BusinessEvents has a default value that provides best performance for most use cases, you can adjust the value to meet your specific needs.</p> <p>The default value is 0(zero), which indicates that prefetch is disabled.</p>
<code>be.engine.cluster.as.member.timeout</code>	<p>The timeout parameter specifies how many milliseconds DataGrid will wait for a member to reconnect, if it loses connection to the metaspace. The default value is 30000 milliseconds.</p>
<code>be.engine.channel.as.querylimit</code>	<p>You can control the query limit for an ActiveSpaces channel using the <code>querylimit</code> property. You can set this property in CDD so that the channel can receive entries beyond 10000 (default in TIBCO ActiveSpaces).</p> <p>The default value in TIBCO BusinessEvents is -1, which indicates no limit on queries.</p>

Property	Notes
<code>be.engine.cluster.as.suspend.threshold</code>	
	The threshold parameter specifies the number of host connections that can be lost before the cluster moves into a suspended state. When the cluster is suspended, members cannot leave or join the cluster. If connectivity is lost for a seeder member of a space, doing a read or write for the space might cause a protocol timeout. The default value is -1, which indicates that the cluster is never suspended.
<code>be.engine.cluster.as.file.sync.interval</code>	
	The amount of time (in milliseconds) to wait between persists to the data store when asynchronous shared-nothing persistence is used. The set value can be viewed as the File Sync Interval property value in <code>as-admin</code> . The default value is 10000 milliseconds.
<code>be.engine.cluster.as.discover.url</code>	
	The discover URL specifies how an engine (node) listens for discovery requests from nodes attempting to join the cluster. PGM protocol is supported for multicast discovery. TCP protocol is supported for unicast (well-known address) discovery. Configuration is different for multicast and unicast discovery. See DataGrid Discover URL for details. The default value for multicast equates to: <code>tibpgm://7888/;239.8.8.9/</code>
<code>be.engine.cluster.as.hostaware.enable</code>	
	By default, this property is true (or enabled). <ul style="list-style-type: none"> If <code>true</code>, the ActiveSpaces member name will be set as: <code>hostname.be-engine-name</code> where <code>be-engine name</code> is what is given on the <code>-n</code> command line option. When host-aware replication is enabled, if the cache nodes are not deployed on multiple machines to satisfy replication by the host, then replication will not happen (or will happen only according to the number of hosts available). For example, if Number of Backup Copies is set to "1" and all cache nodes are deployed on a single host, then replication will be disabled (regardless of the number of cache nodes on that single host). If Number of Backup Copies is set to "2", and cache nodes are deployed only on 2 hosts, then only "1" backup copies will be maintained. If the property is <code>false</code>, host-aware replication will be disabled and the ActiveSpaces member name will be set as <code>be-engine-name</code>. Disabling host-aware replication will honor Number of Backup Copies, provided that there are enough cache nodes deployed in the cluster.
<code>be.engine.cluster.as.hostaware.hostname</code>	

Property	Notes
	Hostnames that are used in identifying members (and therefore naming Shared nothing file/folders), are generated from underlying OS. If you would like to assign hostnames manually instead, for reasons such as hostname/machine changes, testing so on, then provide hostnames in the CDD at each PU level using <code>be.engine.cluster.as.hostaware.hostname</code> property.
<code>be.engine.cluster.as.listen.url</code>	
	<p>The listen URL is used for direct communication between the members of the metaspace after the discovery process. The listen URL uses this format:</p> <pre>tcp://interface:port/</pre> <p>You can also use an auto-incrementing feature by specifying a range as follows:</p> <pre>tcp://interface:Port-[toPort *]/</pre> <p>The default value for <i>interface</i> is the first available interface provided by the operating system for the machine. See DataGrid Listen URL for details.</p> <p>The default value for port is the first available port in the 50000+ range.</p>
<code>be.engine.cluster.as.remote.listen.url</code>	
	<p>Specifies on which IP address and TCP port this proxy metaspace member is listening for the remote client connections. The remote listen URL uses the following format:</p> <pre>be.engine.cluster.as.remote.listen.url=tcp://interface:port</pre>
<code>be.engine.cluster.as.remote.member.timeout</code>	
	<p>Specifies the timeout for remote clients. This is the duration for which the cluster waits for a remote member to reconnect after it got disconnected. If the remote member does not reconnect within this duration, the remote member is considered as disconnected from the cluster.</p> <p>The default value is 120000, that is, 2 minutes.</p>
<code>be.engine.cluster.as.minSeeders</code>	
	TIBCO BusinessEvents sets the value of minimum seeders for user-defined spaces to the same value as the quorum size, by default. To override the default value of minimum seeders, you can specify the new value using this property.
<code>be.engine.cluster.as.node.retry.times</code>	
	<p>Specifies the number of times TIBCO BusinessEvents retries a put or putAll call on the TIBCO BusinessEvents DataGrid cache. Each retry is done after 5 seconds. The number of retries depends on the Lock Timeout property. To calculate the value for retry times, use the following formula:</p> <pre>retry.times=lock.ttl/5+1</pre> <p>For example, if you set Lock Timeout to 30000, then it is recommended that you set the <code>be.engine.cluster.as.node.retry.times</code> value to 7.</p>
<code>be.engine.cluster.as.shutdown.wait.millis</code>	

Property	Notes
	Specifies time (in milliseconds) to wait for the thread that uses ActiveSpaces to complete before shutdown.
<code>be.engine.cluster.minCacheServers.strict</code>	
	<p>When this property is set to true and if the number of Cache nodes drops below Quorum, then the cluster is placed into suspend mode.</p> <div>  <p>When the system is actively processing messages, it may take a while for it to reach the suspended state as the Inference agents attempt to commit current transactions and empty the internal queues before suspending the operations.</p> </div> <p>Default value is false.</p>
<code>be.engine.cluster.minCacheServers.strict.selfRepair</code>	
	<p>When this property is set to true, the system tries to resume operations if and only when the Quorum is reached again. If this property is set to false once the operations are suspended, then the system will stay in that mode until you manually intervene. Default value is false.</p> <div>  <p>This property only applies if <code>be.engine.cluster.minCacheServers.strict=true</code>.</p> </div>
<code>be.engine.cluster.quorumCheck.setLenient</code>	
	<p>Using this property, you can change the quorum state behavior. The values are:</p> <ul style="list-style-type: none"> <code>true</code> - For the initial startup, a minimum number of cache nodes are required as specified in the quorum count. The quorum state of the cluster is maintained after a quorum is established until the number of cache nodes drop below the minimum number of seeders as specified in the <code>be.engine.cluster.as.minSeeders</code>. During this period, new cache nodes can join the cluster and function like the quorum is never lost. The new Inference engine can be started as long as the number of cache nodes is not below the minimum number of seeders. <code>false</code> - For the initial startup, a minimum number of cache nodes are required as specified in the quorum count. After a quorum is established, the system keeps running till the number of cache nodes drop below the minimum number of seeders as specified in the <code>be.engine.cluster.as.minSeeders</code>. Now the new inference agents cannot be started until the number of cache nodes reaches the quorum count. <p>The default value is <code>false</code> for inference engines, and <code>true</code> for cache engines.</p> <p>This property can be used at the agent level as well.</p>
<code>be.engine.cluster.cacheNaming.isDescriptive</code>	

Property	Notes
	<p>Specifies whether descriptive information (such as dist/repl, unlimited/limited, nob/bs, and so on) is included in the shared nothing cache names. The values are:</p> <ul style="list-style-type: none"> • <code>true</code> - names are descriptive. For example, <code>C:\temp\sharednothing\TestBQL\dist-unlimited-bs-TestBQL--be_gen_Concepts_Agreement\CSU1\CSU1_store_1440456000</code>. • <code>false</code> - names are not descriptive. For example, <code>C:\temp\sharednothing\TestBQL\TestBQL--be_gen_Concepts_Agreement\CSU1\CSU1_store_1440456000</code>. <p>The default value is <code>true</code>.</p>
<code>be.engine.cluster.as.security.mode.role</code>	
	<p>Security role of a node for the secure DataGrid. Possible values are: <i>Controller</i> or <i>Requestor</i>.</p> <p>The Controller is dedicated to enforcing security behavior for a cluster associated with the security domain. Security Controllers are the only discovery nodes in a cluster.</p> <p>The Requestor requires access to the data in the DataGrid, which needs to be authorized by the Controller. A Requestor can never be used as a discovery node.</p>
<code>be.engine.cluster.as.security.file</code>	
	<p>Path to the policy (for controller) or token file (for requestor), which contains the security settings, based on role defined in the <code>be.engine.cluster.as.security.mode.role</code> property.</p>
<code>be.engine.cluster.as.security.file.identity.password</code>	
	<p>The password for the identity key in the security policy file or token file specified in <code>be.engine.cluster.as.security.file</code>.</p>
<code>be.engine.cluster.as.security.requester.identity.keyfile</code>	
	<p>The absolute path for a file containing the key to use for LDAP with the certificate-based authentication.</p>
<code>be.engine.cluster.as.security.domain</code>	
	<p>Optional. Domain name for system based user authentication.</p>
<code>be.engine.cluster.as.security.username</code>	
	<p>User name for LDAP and system based authentication.</p>
<code>be.engine.cluster.as.security.password</code>	

Property	Notes
	<p>Password for LDAP and system based authentication. In case authentication type in the policy file is "x509" then this is the password is for the private key in the LDAP identity file specified in <code>be.engine.cluster.as.security.requester.identity.keyfile</code>.</p>
<code>be.mm.cluster.as.listen.url</code>	
	<p>If you use well-known-address discovery and TIBCO BusinessEvents Monitoring and Management (MM), you must also add this property at the cluster level of the to-be-monitored project's CDD file.</p> <p>Specify the value as the IP of the computer hosting the MM server and an unused port.</p> <p>See Cluster Discovery and Internal Communication for the procedure, including an additional step you must take.</p>
<code>be.engine.cluster.as.remote.tuple.limit</code>	
	<p>This property controls the number of entries or records sent from seeder to client. When volume of data in cache gets very high then querying the spaces results into blocking threads, in such case, you can add this property to send only limited records from seeders to the client. This is generic property which can be applied to all agents including the remote client. The valid values are any positive long numbers and -1.</p> <p>The default value is -1 which indicates no limit on entries or records.</p>
<code>be.engine.cluster.event.expiry.lock</code>	
	<p>Set this property to true to enable cluster-level locking on event extID for the event expiry thread.</p> <p>The default value is false.</p>
<code>be.engine.cluster.as.extid.index</code>	
	<p>Set this property to true to add an index for extID.</p> <p> The default value is true for the Shared Nothing option.</p>
<code>be.engine.cluster.as.browser.join</code>	
	<p>Set this property to true to enable the space join on browse calls during processing.</p>
<code>be.engine.cluster.as.browser.timeout</code>	
	<p>Use this property to configure the timeout for an ActiveSpaces browser. Provide the timeout in milliseconds.</p>
<code>be.engine.cluster.as.invocation.timeout</code>	
	<p>Use this property to configure the timeout for an ActiveSpaces member invocation. Provide the timeout in milliseconds.</p>

Enabling Use of Oracle Coherence as the Cache Provider

If you want to use the Oracle Coherence cache provider, you must provide a fully licensed, supported version of the software.

See the product readme file for supported version information. You must also enable Oracle Coherence as the cache provider, as explained below.



You cannot use the `coherence.jar` file from an earlier TIBCO BusinessEvents release.

Procedure

1. Copy the `coherence.jar` file from your Oracle Coherence installation to `BE_HOME/lib/ext/tcpl`.
(This location is preconfigured in the `studio/eclipse/configuration/studio.tra` classpath, as shipped. If you use a different location, update the classpath.)
2. Enable the Coherence category functions as follows:
 - a) Open the `BE_HOME/studio/eclipse/configuration/studio.tra` file for editing.
 - b) Change the setting of the following property to true (it is false as shipped):

```
TIBCO.BE.function.catalog.coherence=true
```
 - c) Save the file and restart TIBCO BusinessEvents Studio.

The log messages printed to the console at startup show the location of the files in use. See `BE_HOME/bin/logs/cep-engine.log`.

Oracle Coherence Cluster Discovery

You can use either multicast or well-known-address (WKA) discovery, as appropriate.

See Cache Cluster Member Discovery in *TIBCO BusinessEvents Architect's Guide* for basic guidelines. Procedures for both methods of discovery are explained below.

See [Enabling Use of Oracle Coherence as the Cache Provider](#) for prerequisite actions you must take before you can use Oracle Coherence as the cache provider.

Guidelines for Managing Coherence Clusters

To manage Coherence clusters refer to the provided guidelines.

The following files are located in `BE_HOME/lib/cep-datagrid-oracle.jar`:

- `coherence-cache-config-jdbc.xml` is an example cache configuration descriptor file.
- `tangosol-coherence-override-tibco-be.xml` is an example operational descriptor override file. To reference such a file, use the property `tangosol.coherence.override` in the CDD file.

To understand when and how to use them, read the Coherence documentation.

The following links to Coherence documentation provide helpful information for use of Coherence as the cache provider.

Checklist and Guidelines Before Architecting a New Project

- http://download.oracle.com/docs/cd/E15357_01/coh.360/e15723/deploy_checklist.htm
- http://download.oracle.com/docs/cd/E15357_01/coh.360/e15723/deploy_plat_consider.htm
- http://download.oracle.com/docs/cd/E15357_01/coh.360/e15723/tune_perftune.htm

Coherence Network Protocol

- http://download.oracle.com/docs/cd/E15357_01/coh.360/e15723/cluster_tmp.htm
- http://download.oracle.com/docs/cd/E15357_01/coh.360/e15723/tune_datagramtest.htm

Coherence Metrics

- http://download.oracle.com/docs/cd/E15357_01/coh.360/e15723/appendix_mbean.htm
- http://download.oracle.com/docs/cd/E15357_01/coh.360/e15723/appendix_operational.htm

Configuring Multicast Cluster Discovery for Coherence Clusters

Multicast is the default option. If default values for multicast properties work in your environment, you can omit this procedure.

To Configure Multicast Cluster Discovery

- If the default values for the following properties are not appropriate for your environment, add them to the properties sheet of the CDD Cluster tab and specify values as needed:

```
tangosol.coherence.clusteraddress
tangosol.coherence.clusterport
tangosol.coherence.ttl
```

See [Multicast Discovery Properties](#) for details

- For multicast discovery you may also need to set these properties as explained in [Localhost and Localport Properties](#):

```
tangosol.coherence.localhost
tangosol.coherence.localport
```



Specifying one or more well-known addresses disables all multicast communication. Remove any well-known address properties, if any.

Configuring Well-Known Address Cluster Discovery

You can add up to nine well-known addresses. If you need to add more refer to Coherence documentation for instructions about using override files.

To configure well-known address discovery, you must configure machine-specific settings at the cluster level and at the processing unit level. One processing unit deployed to a WKA machine must have the additional WKA configuration. Additional processing units can be deployed to a WKA machine, configured in the usual way, and they will discover and join the cluster in the usual way at runtime.

For details about the properties, see [Well-Known Address Properties](#).

Procedure

1. In the CDD file editor Cluster tab properties sheet, add a pair of WKA properties for each machine you want to configure as a well-known address machine:

```
tangosol.coherence.wkan HostIP
tangosol.coherence.wkan.port Hostport
```

For example at the cluster level you might have these two WKA machines:

Property: tangosol.coherence.wka1	Value: 10.97.118.151
Property: tangosol.coherence.wka1.port	Value: 8098
Property: tangosol.coherence.wka2	Value: 10.97.118.152
Property: tangosol.coherence.wka2.port	Value: 8098



If two engines run on one machine (with a single IP), ensure that each engine uses a different port. For example:

Property: <code>tangosol.coherence.wka1</code>	Value: <code>10.97.118.151</code>
Property: <code>tangosol.coherence.wka1.port</code>	Value: <code>8098</code>
Property: <code>tangosol.coherence.wka2</code>	Value: <code>10.97.118.151</code>
Property: <code>tangosol.coherence.wka2.port</code>	Value: <code>8099</code>

2. In the Processing Units tab properties sheet for a processing unit (PU), configure one set of WKA properties to match one of the cluster level set of WKA properties:

```
tangosol.coherence.localhost HostIP
```

```
tangosol.coherence.localport Hostport
```



If you will deploy using TIBCO BusinessEvents Monitoring and Management (MM) also add the following property:

```
be.engine.hostaddress HostIP
```

Use the same the value as the value of the `tangosol.coherence.localhost` property. See [Configuring for Coherence WKA Cluster Discovery in , Basic MM Configuration in TIBCO BusinessEvents Administration](#) for more steps required when configuring MM.

Set the `localhost` property to the IP of the host where you will deploy the PU, and set the `localport` property to the port defined in the cluster properties `localport` property.

3. Repeat the previous step until you have configured a PU with matching PU-level properties for each cluster-level (that is, machine-level) set of WKA properties.

For example, if you configured the two well-known addresses, then at the processing unit level you would configure two processing units as follows:

```
tangosol.coherence.localhost. Value: 10.97.118.151
tangosol.coherence.localport. Value: 8098
```

```
tangosol.coherence.localhost. Value: 10.97.118.152
tangosol.coherence.localport. Value: 8098
```

At deploy time you must deploy the processing units on the appropriate (matching) WKA machine. It can be helpful if the name of the PU contains the machine name as a reminder that this PU must be deployed to that machine.

See [Localhost and Localport Properties](#).

CDD Cluster Tab Coherence Properties Reference

Coherence properties can be configured using the CDD Cluster Tab. Refer to the following table for a list of properties that you can use to retrieve data from the coherence properties.



Oracle Coherence Cache Provider


This section is relevant only if you use Oracle Coherence as the cache provider.






Add properties as needed to configure multicast cluster member discovery, or well-known address cluster member discovery. See [Oracle Coherence Cluster Discovery](#) for details.

CDD Cluster Tab Coherence Properties

Property	Notes
Multicast Discovery Properties	
If you will define cluster members using multicast discovery properties, use the properties in this section, instead of those in the section Well-Known Address Properties .	

Property	Notes
<code>be.engine.cluster.as.minSeeders</code>	
	TIBCO BusinessEvents sets the value of minimum seeders for user-defined spaces to the same value as the quorum size, by default. To override the default value of minimum seeders, you can specify the new value using this property.
<code>be.engine.cluster.minCacheServers.strict</code>	
	<p>When this property is set to true and if the number of Cache nodes drops below Quorum, then the cluster is placed into suspend mode.</p> <div>  <p>When the system is actively processing messages, it may take a while for it to reach the suspended state as the Inference agents attempt to commit current transactions and empty the internal queues before suspending the operations.</p> </div> <p>Default value is false.</p>
<code>be.engine.cluster.minCacheServers.strict.selfRepair</code>	
	<p>When this property is set to true, the system tries to resume operations if and only when the Quorum is reached again. If this property is set to false once the operations are suspended, then the system will stay in that mode until you manually intervene. Default value is false.</p> <div>  <p>This property only applies if <code>be.engine.cluster.minCacheServers.strict=true</code>.</p> </div>
<code>tangosol.coherence.clusteraddress</code>	
	<p>Use this setting if multicast discovery is used and if you need a non-default value. Specifies the multicast IP address that the socket will listen to or publish on.</p> <p>Possible values are addresses between (and including) 224.0.0.0 and 239.255.255.255.</p> <p>Default value is 224.3.3.1</p>
<code>tangosol.coherence.clusterport</code>	
	<p>Use this setting if multicast discovery is used and if you need a non-default value. Specifies the port that the socket will listen to or publish on.</p> <p>Possible values are integers between 1 and 65535.</p> <p>Default value is 35463.</p>
<code>tangosol.coherence.ttl</code>	

Property	Notes
	<p>Specifies the time-to-live setting for the multicast, that is, the maximum number of "hops" a packet can traverse. A hop is defined as a traversal from one network segment to another via a router.</p> <p>For production use, set this value to the lowest integer value that works. Setting the value too high can use unnecessary bandwidth on other LAN segments and can even cause the operating system or network devices to disable multicast traffic.</p> <p>Set the single-host cluster to the value of 0 (zero). Set the simple switched backbone to the value of 1 (one). Set the advanced backbone with intelligent switching to the value of t may require a value of 2 (two) or more.</p> <p>A value of 0 is intended to keep packets from leaving the originating machine. However, some operating systems do not implement this correctly, and the packets may in fact be transmitted on the network. It is required for multicast configuration.</p> <p>Possible values are integers between 0 and 255.</p> <p>Default value is 4.</p>
Localhost and Localport Properties	
Use the <code>localhost</code> and <code>localport</code> properties for the following cases:	
<ul style="list-style-type: none"> • When a host has multiple network cards. • For multicast discovery when more than one cluster is running on the same subnet (localhost is required but not localport in this case). 	
Add these properties at the PU level when well-known address discovery is used, as explained in Configuring Well-Known Address Cluster Discovery .	
Default values are provided at the cluster level. However if you need to specify these properties at the PU level, add them as Processing Units tab properties and provide the values as needed.	
<code>tangosol.coherence.localhost</code>	
<p>Specifies the IP address that the socket will listen to or publish on.</p> <p>As needed, you can set the value of the <code>localhost</code> property to the value <code>localhost</code>. However, if <code>localhost</code> is used as the loop-back address (127.0.0.1) you must enter a machine name or IP address.</p> <p>Default value is <code>localhost</code>.</p>	
<code>tangosol.coherence.localport</code>	

Property	Notes
	<p>Specifies the port that the socket will listen to or publish on.</p> <p>Possible values are 1 to 65535.</p> <p>Default value is 8088.</p> <div>  <p>If a specified port is not available, the object management layer (by default) increments the port number until it finds an available port. Avoid potential conflicts by choosing a number that is not close to a port used by other software in your environment.</p> </div> <div>  <p>To turn off auto-incrementing, add the following property: <code>tangosol.coherence.localport.adjust=false</code></p> </div>
<p>Well-Known Address Properties</p> <p>See Configuring Well-Known Address Cluster Discovery .</p> <div>  <p>If you discover cluster members using well-known addresses, use the properties in the referenced section and remove the multicast discovery properties shown in the section Multicast Discovery Properties .</p> </div>	
<code>tangosol.coherence.wkan.port</code>	
<p>The following are addresses and ports for machines used by the well-known address cluster discovery protocol.</p> <p>At least one of these machines must be running at any time so that others can join the cluster.</p> <p>For <code>tangosol.coherence.wkan</code>, enter the IP address.</p> <p>For <code>tangosol.coherence.wkan.port</code>, enter a value between 1 and 65535.</p> <p>For example (in the UI the properties are not entered quite this way):</p> <pre>tangosol.coherence.wka1 10.97.118.151 tangosol.coherence.wka1.port 8088 tangosol.coherence.wka2 10.97.118.152 tangosol.coherence.wka2.port 8088</pre> <p>Also at the Processing Units tab, configure <code>localhost</code> and <code>localport</code> properties for one processing unit that will be deployed to the WKA machine. Set the <code>localhost</code> value to the value of the <code>wkan</code> property. Set the <code>localport</code> value to the value of the <code>wkan.port</code> property. (See Localhost and Localport Properties)</p> <div>  <p>You can configure two well-known addresses for the same machine, and use a different port number for each. In this case you would also configure two processing units, each of which matches one set of WKA properties.</p> </div> <div>  <p>To turn off auto-incrementing, add the following property: <code>tangosol.coherence.localport.adjust=false</code>.</p> </div>	
<p>Other Coherence Properties</p> <p>These properties are used in various situations.</p>	

Property	Notes
<code>tangosol.coherence.distributed.threads</code>	<p>Specifies the number of Coherence domain threads used by the distributed cache service when Oracle Coherence is used as the cache provider.</p> <p>This property is mainly used with write-behind database strategy. However, when used with the cache-aside strategy, this setting is used for handling cache operations only (gets and puts). In this release, the property must be set to the same value across the. The value must be set to <code>processing unit</code>. A value of zero 0 signifies that all of the relevant tasks are performed on the service thread.</p> <p>See Write Behind Options in <i>TIBCO BusinessEvents Architect's Guide</i>.</p> <p>Default value is 0.</p>
<code>tangosol.coherence.override</code>	<p>Specifies the location of an Operational Descriptor Override File. A sample value is: <code>file:/c:/tmp/my_tangosol-coherence-override.xml</code> .</p> <p>An example of an override file is provided in <code>BE_HOME/lib/cep-datagrid-oracle.jar</code>.</p> <p>Use of an operational descriptor override is not generally required. For details on override files, see Coherence documentation.</p>
<code>tangosol.coherence.localport.adjust</code>	<p>An auto-incrementing feature ensures that a different port is used if one specified is already in use. However, in various situations you may want to turn off this behavior. For example, if you use TIBCO BusinessEvents Monitoring and Management, and the MM server runs on the same machine as any of the monitored cluster engines, you must explicitly ensure that all ports used by MM and the monitored cluster are unique. Therefore, the auto-incrementing feature may not be appropriate.</p> <p>To turn off auto-incrementing, add this property <code>tangosol.coherence.localport.adjust</code> and set the value to <code>false</code>.</p> <p>Default value is <code>true</code>.</p>
<code>tangosol.coherence.guard.timeout</code>	<p>When infrastructure latency occurs such as remote databases, you can improve the performance of inference engines by setting the property value to 0 in <code>be-engine.tra..</code></p> <p>For example: <code>tangosol.coherence.guard.timeout=0</code>.</p>

CDD Load Balancer

Agents are configured to work cooperatively as routers and receivers to ensure that related messages arriving from queue sources are handled by the same agent, so that related information is available locally.

Currently, only queue messages from TIBCO Enterprise Message Service are supported for this configuration.

Load Balancing Options

Load balancing is available for messages arriving from queues. Do not use load balancing for topic-based or other broadcast sources.

Two kinds of load balancing configurations are available: basic load-balancing and content-aware load balancing. Both of these configurations support messages arriving from TIBCO Enterprise Message Service queue sources.

Every JMS destination that is configured to be an input destination runs in its own JMS Session. This provides good throughput on queues for processing, and less connections.

Basic Load Balancing

With basic load balancing, events from queue sources are automatically distributed between deployed instances of an agent class. To set up this kind of load balancing, you deploy multiple instances of an agent class that listens to a JMS destination. Each deployed agent instance runs in a different processing unit.

This method can be useful when there is no relationship between the events that would require them to be processed in a certain order. If the order or grouping of events received is important, use content-aware load balancing. Content-aware load balancing has other benefits also, as explained below.

Content-Aware Load Balancing

With content-aware load balancing, all related events arriving from queues are routed to the same agent using a routing key.

The key is formed using the event properties (single or multiple). For example, if the event property is `ZipCode` then a routing key is a specific zip code. All messages relating to one zip code are routed (over TCP) to the same agent, providing "session stickiness."

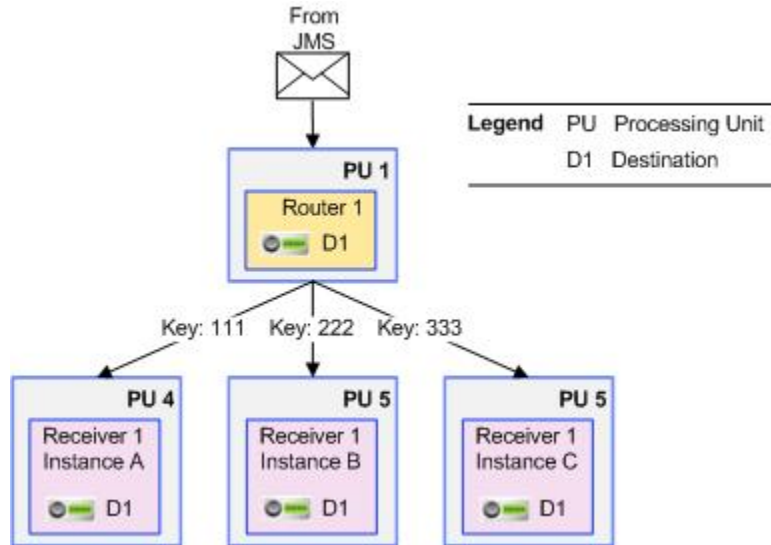
Content-aware load balancing uses *routers* and *receivers*. You can configure routers and receivers from the CDD Load Balancer tab. One receiver can handle more than one set of related events. For example if the routing key is a zip code, one receiver might handle events for multiple zip codes.

Use of content-aware load balancing simplifies project configuration, and makes runtime behavior more efficient. For example, only local locking is generally required (whereas basic load balancing requires cluster-wide locking). Also the L1 cache does not have to be checked for version consistency.



The JMS message acknowledgment modes `EXPLICIT_CLIENT_ACKNOWLEDGE` and `EXPLICIT_CLIENT_DUPS_OK_ACKNOWLEDGE` are only supported.

CDD Based Router and Receiver Configuration



Routers

A router PU receives messages from the JMS server and routes them to appropriate receivers. Routers do no other work. For example, they should not execute rules.

A router PU contains one inference agent with one or more sets of JMS channels and destinations. Each destination has a default event. Values of one property or combination of multiple properties of that event are used at runtime as *routing keys*.

Event preprocessors can be used as needed to populate the routing key property, for example using some calculation or combination of other event properties.

The router redirects events over TCP to a receiver, based on the destination and the routing key values. The router transparently distributes the load across the available receivers. If a receiver agent fails, its messages (that is, messages with the key that the router was sending to that agent) are routed to another receiver and continue to be handled by that other receiver.

You can run multiple router instances using the same routing configuration, and they will all follow the same routing strategy. If one of the router fails, the one that is running will continue with routing process. No configuration is required for routers to work in fault tolerance mode as they do it intrinsically.

Receivers

Receivers are the inference or query agents that do the actual work. A receiver PU contains one inference or query agent. A set of receivers belongs to the same agent class. Receivers can also do other work, in addition to the work they receive from the router.

With CDD configuration, a receiver agent class is configured with one of the channel and destination configurations defined in the router. The destination, however, functions as a pseudo destination.

Content-Aware Load Balancer

When setting up content-aware load balancing, you first choose between two methods: pair configuration and adhoc configuration.

Pair Configuration

Preconfigured using the Load Balancer tab of the CDD.

The pair configuration uses a pair of processing units to act as routers and receivers. Two agent classes, a receiver inference class and a router inference class, are configured to use the same EMS destination.

The router processes messages from the EMS queue and sends it to a receiver based on the routing key specified in the CDD.

See [CDD Load Balancer Tab Properties Reference](#) for details.

Adhoc Configuration

Allows minimal preconfiguration using the Load Balancer tab of the CDD. Use catalog functions to implement the load balancer at runtime.

Only the load balancer name and the local destination are configured in the CDD. The router agent and receiver agent classes are configured using catalog functions in their rules and rulefunctions.



Ensure that the destination has a default event associated with it.

Adhoc Load Balancer

Guidelines are provided to configure the router and receivers in an adhoc load balancer configuration.

Router Configuration

`LoadBalancer.Router.*` functions are used for the router side.

Receiver Membership Functions in Catalog

`LoadBalancer.Receiver.Membership.isInFlux`

Returns true if the loadbalancer node membership is in a state of change such as nodes joining and/or leaving currently or in the recent past.

`LoadBalancer.Receiver.Membership.getRecentChangeAt`

Returns the timestamp (1970 epoch milliseconds) at which the most recent membership change occurred.

Creating the Load Balancer

The startup rule function has to be provided. It creates and returns a load balancer that can be used to send messages to load balanced remote destinations.

Procedure

1. Create the router TCP connection in a startup rule function.
`LoadBalancer.Router.createLoadBalancerTo(adhocConfigName);`
2. Send Event to the Receiver .
Use this rule function as an event preprocessor. It sends an event to a remote receiver. The router agent does not have any destinations. The routing decision is made using the routing key.

```
void LoadBalancer.Router.send(Object loadBalancer , SimpleEvent event, String routingKey);
```

3. Discard the Load Balancer.
Put this rule function in a shutdown rule function.

```
Object loadBalancer = Util.HashMap.remove(String mapID, String key);
Util.HashMap.deleteMap(String mapID);
```

4. This rulefunction discards the load balancer.

```
void LoadBalancer.Router.discardLoadBalancer(Object loadBalancer);
```

Configuring the Receiver

Configuring a Receiver consists of obtaining information, creating the Receiver, creating the Receiver TCP connection, and discarding the Receiver.

Procedure

1. Use the following function for the Receiver side:

```
LoadBalancer.Receiver.*
```
2. Obtain information for a local channel needed for the Receiver:

```
int port = System.getPropertyAsInt("receiver_localchnl_localdest_port",
34567);
```
3. Create a Receiver.
 The receiver object, which receives messages from a router, is created and returned. Messages will be received from the router on the local channel and destination specified.
4. Create the Receiver TCP connection in a startup rule function.

```
Object LoadBalancer.Receiver.createTcpReceiverFor(String adhocConfigName
```
5. Discard the Receiver in the shutdown rule function.

```
LoadBalancer.Receiver.discardReceiver(Object loadBalancedReceiver;
```

CDD Load Balancer Tab Properties Reference

A load balancer is configured by specifying the receiver and router agents, and setting the destination. To configure the load balancer, use the CDD Load Balancer tab.

CDD Load Balancer Tab Properties

Property	Notes
Pair Configuration	
Name	Name of the pair configured load balancer.
JMS Destination	Destination used by the router and receiver agents. Ensure that the destination has a default event configured.
Key	Routing key used by the pair configuration.
Router	Router agent class for the load balancer configuration.
Receiver	Receiver agent class for the load balancer configuration.
Adhoc Configuration	
Name	Name of the adhoc load balancer. When creating a load balancer using the catalog function <code>createLoadBalancerTo</code> , the name of the load balancer must be specified. For example, <pre>Object loadBalancer = LoadBalancer.Router.createLoadBalancerTo("adhocLoadBalancerName");</pre>

Property	Notes
JMS Destination	Destination used by the router and receiver agents. In an adhoc configuration, a local channel with a local destination is used to communicate between the router and receiver agents.
Properties	
transport	Transport to enable communication between the receiver and the router. Typically, this happens through an internal TCP connection. The default value is <code>tcp</code> .
hostname	Hostname where the load balancer is configured. The default value is <code>localhost</code> .
port	Port number used by the specified transport.



The `transport`, `hostname`, and `port` properties must be in lowercase.

CDD Backing Store

To provide for data persistence, you can implement a backing store for use with Cache OM.

During regular operation, cache data is written to the backing store. On system restart, data in the backing store is restored to the cache cluster.

To implement a backing store, provide a supported database product. Scripts are provided to set up the database for your project's ontology. If the ontology changes, scripts help you adapt the backing store accordingly and existing backing store data can be preserved.

Oracle Database Strategy

If the Strategy field for an Oracle database with a Shared All persistence (set under the Cluster tab > Backing Store) is set to `oracle`, then Oracle Database pooling strategy settings are used and various CDD properties act on the corresponding Oracle property, as noted in the reference table.



In some circumstances, it is necessary to configure some backing store settings before you set up the backing store. See [Ontology Identifiers that Exceed the DBMS Maximum Column Length](#).

Configuring Backing Store Settings and Properties

The Cluster tab is used to configure the backing store settings and properties.

Procedure


1. Open the CDD file you added.
2. Select the Cluster tab > Backing Store node on the left and on the right, configure settings as explained in [CDD Cluster Tab Backing Store Settings Reference](#).
3. Select the Cluster tab > Connection node on the left and on the right configure database connection settings. See [Backing Store > Connection Settings](#).
4. Select the Cluster tab Properties node on the left and configure properties on the right, following guidelines in [CDD Cluster Tab Backing Store Properties Reference](#).
5. You can select domain objects (entities) to be included in or excluded from the backing store. In addition preloading options are available for loading domain objects from backing store to cache at system startup. See [Domain Objects Configuration](#).
 - See [Preloading Options](#) for object settings
 - See `be.engine.cluster.recovery.threads` in [CDD Cluster Tab Backing Store Properties Reference](#).
6. Save the resource.

CDD Cluster Tab Backing Store Settings Reference

Use this reference for the backing store settings.

For the related procedure, see [Configuring Backing Store Settings and Properties](#).

CDD Cluster Tab Backing Store Settings

Property	Notes
Persistence Option: None <p>Specifies that the cluster does not have a backing store or the backing store is temporarily disabled, for example during testing phases of a project.</p> <div>  <p>Individual entities can be set to not use the backing store. See Has Backing Store in CDD Cluster Tab Domain Object Override Settings Reference .</p> <p>Default is set to <code>None</code>.</p> </div>	
Persistence Option: Shared All	
Database Type	<p>Select which of the supported DBMS products to use:</p> <ul style="list-style-type: none"> • Oracle • SQL Server • DB2 • MySQL • Berkeley DB. • Postgres <p>The default value is <code>Oracle</code>.</p>
Strategy	<p>Used only if the Database Type is set to <code>Oracle</code>.</p> <p>If you use Oracle Database, you have the option of using either the TIBCO BusinessEvents internal pooling implementation, or Oracle Database's implementation. Possible values are as follows:</p> <ul style="list-style-type: none"> • <code>jdbc</code> - Use the internal pooling mechanism. • <code>oracle</code> - Use Oracle's pooling mechanism (see the class <code>OracleConnectionCacheManager</code> in the package <code>oracle.jdbc.pool</code>). When set to <code>oracle</code> then the TIBCO BusinessEvents pooling property values are used to set their corresponding to Oracle Database properties. <p>The default value is <code>jdbc</code>.</p>

Property	Notes
Cache Aside	<p>Not available for Berkeley DB.</p> <p>Choose between these two options:</p> <ul style="list-style-type: none"> • Checkbox unchecked means Write Behind- Writes data to the cache and then to the backing store. One write-behind thread is used for each entity type. If write-behind strategy is used with Oracle Coherence cache provider, you can also set <code>tangosol.coherence.distributed.threads</code> in the cluster level properties. • Checkbox checked means Cache Aside - Writes data to the cache and at the same time to the backing store. User controls are available for threads and queue size, and other options such as using parallel or sequential operations in the post-RTC phase. See Threading Models and Tuning in <i>TIBCO BusinessEvents Architect's Guide</i> for more information.
Enforce Pools	<p>Not available for Berkeley DB.</p> <p>Check this property if you want to enforce connection pool properties. See Database Connection Properties for the property details.</p> <p>Default is unchecked.</p>
Berkeley DB Data Store Path	<p>Available only for the backing store database type Berkeley DB.</p> <p>Enter the absolute path of the data store to be used. The data store needs to be located on a network drive that can be accessed by all the cache nodes.</p>
Persistence Option: Shared Nothing	
Persistence Path	<p>Specifies the absolute path to the directory where the data is to be stored. For example, <code>/tmp/datastore/</code>.</p> <p>Set this value for each individual processing units.</p>
Persistence Policy	<p>Specifies the type of communication to be used to maintain persistence: asynchronous (ASYNC) or synchronous (SYNC).</p> <ul style="list-style-type: none"> • ASYNC - This policy is recommended if you want to avoid frequent IO operations, which can slow inference agents. • SYNC - Solid State Drives (SSD) are recommended when using this policy. <p>Default is <code>async</code>.</p>
Backing Store > Connection Settings	
<p>Try running with default pool values and monitor the behavior. Using more connections improves runtime performance and can also speed up recovery in the event of a failure.</p> <p>Pool settings are used only if Enforce Pools is checked.</p>	

Property	Notes
URI	<p>Specifies the project path, that is, the path from the project root to the JDBC Connection resource, to define the connection to the backing store. For example:</p> <pre>/SharedResources/JDBC Connection.sharedjdbc</pre> <p>You can also use a global variable to specify the connection.</p> <p>Default value is <code>%%DbUri1%%</code>.</p>
Min Size	<p>Minimum number of JDBC connections in the JDBC connection pool used for the backing store.</p> <p>Oracle Database Strategy</p> <p>If the Cluster tab > Backing Store > Strategy field is set to <code>oracle</code>, then Oracle Database strategy settings are used, and this property corresponds to the <code>OracleConnectionCacheManager</code> class property <code>MinLimit</code>.</p> <p>Default is 10.</p>
Max Size	<p>Maximum number of JDBC connections in the JDBC connection pool used for the backing store. Connections do not exceed the maximum.</p> <p>The value of this property overrides the value of the Maximum Connections setting in the JDBC Connection resource.</p> <p>Although the limit is seldom reached, you can guarantee a connection is always available for a <code>dbwriter</code> thread as follows. Set this field to the same value as the <code>Agent.AgentClassName.dbthreadcount</code> setting.</p> <p>Similarly (and also seldom needed), with Coherence cache provider, you can guarantee a connection is available by setting this field to the same value as the property <code>tangosol.coherence.distributed.threads</code>.</p> <p>Oracle Database Strategy</p> <p>If the Cluster tab > Backing Store > Strategy field is set to <code>oracle</code>, then Oracle Database strategy settings are used, and this property corresponds to the <code>OracleConnectionCacheManager</code> class property <code>MaxLimit</code>.</p> <p>Default is 10.</p>
Initial Size	<p>Specifies the initial size of the JDBC connection pool used for the backing store, when it is created on startup. For example:</p> <p>Oracle Database Strategy</p> <p>If the Cluster tab > Backing Store > Strategy field is set to <code>oracle</code>, then Oracle Database strategy settings are used, and this property corresponds to the <code>OracleConnectionCacheManager</code> class property <code>InitialLimit</code>.</p> <p>Default is 10.</p>

CDD Cluster Tab Backing Store Properties Reference

Use this reference for the backing store properties.

For the related procedure, see [Configuring Backing Store Settings and Properties](#).

CDD Cluster Tab Backing Store Properties

Property	Notes
Database Connection Properties	
Used only if Enforce Pools is checked (see CDD Cluster Tab Backing Store Settings Reference).	
<code>be.backingstore.dburi.pool.waitTimeout.0</code>	
	<p>Used only if the Strategy setting (see CDD Cluster Tab Backing Store Settings Reference) is set to <code>oracle</code>.</p> <p>Oracle Database Strategy</p> <p>If the Cluster tab > Backing Store > Strategy field is set to <code>oracle</code>, then Oracle Database strategy settings are used, and this property corresponds to the <code>OracleConnectionCacheManager</code> class property <code>WaitTimeout</code>.</p> <p>Specifies behavior when a connection is requested and there are already Cluster tab > Backing Store > Connection > Max Size connections active. If the <code>be.backingstore.dburi.pool.waitTimeout.0</code> value is greater than zero (0), each connection request waits for up to the specified number of seconds. If no connection is returned to the pool before the timeout elapses, a <code>No Database Connection available</code> exception is thrown.</p> <p>The <code>waitTimeout</code> and <code>inactivityTimeout</code> properties specify wait periods to minimize the creation and destruction of connections (an expensive operation).</p> <p>Default is 1 second.</p>
<code>be.backingstore.dburi.pool.inactivityTimeout.0</code>	
	<p>Oracle Database Strategy</p> <p>If the Cluster tab > Backing Store > Strategy field is set to <code>oracle</code>, then Oracle Database strategy settings are used, and this property corresponds to the <code>OracleConnectionCacheManager</code> class property <code>InactivityTimeout</code>.</p> <p>Specifies the number of seconds an unused connection remains available (so that other threads can use it). After this period, the connection is closed and removed from the pool.</p> <p>Default value is 900 seconds.</p>
<code>be.backingstore.readtimeout</code>	

Property	Notes
	<p>Oracle Database Strategy</p> <p>If the Cluster tab > Backing Store > Strategy field is set to <code>oracle</code>, then Oracle Database strategy settings are used, and this property corresponds to the <code>OracleConnectionCacheManager</code> class property <code>ReadTimeout</code>.</p> <p>Use this property to handle situations where engines running inference agents hang when the JDBC connection to the backing store is slow or intermittent. If no response is received from the database within the specified period, a call is aborted.</p> <p>Time unit is milliseconds.</p> <p>A value of 0 (zero) means that no timeout is set.</p> <p>Default value is 0 .</p>
<code>be.backingstore.recreateOnRecovery</code>	
	<p>Set this property to true if the database pool size does not recover to the initial or minimum connection size, as defined by Min Size and Max Size properties (in CDD Cluster Tab Backing Store Settings Reference).</p> <p>Default value is false .</p>
Other properties	
<code>be.backingstore.useobjecttable</code>	
	<p>The property when set to true provides mappings for all entities in the cache. Object table is used to find the actual object either in the cache or in the backing store.</p> <p>When this property is set to false, you must use the catalog functions with the "byURI()" pattern so that entities are found from the cache.</p> <p>Default value is true .</p>
<code>be.backingstore.commitSize</code>	
	<p>Used with the Coherence cache provider and write-behind strategy only.</p> <p>Sets the maximum number of RTC transactions that a distributed cache service thread takes from the distributed cache service queue and processes in one batch. When threads are idle, they take jobs from the queue in smaller batches.</p> <p>Set this property to the desired number of transactions to suit your needs.</p> <p>Refer to the "Write Behind Options" section in the <i>TIBCO BusinessEvents Architect's Guide</i> for more details.</p> <p>Default value is 10.</p>
<code>be.backingstore.optimize.reads</code>	

Property	Notes
	<p>Used with Microsoft SQL Server only.</p> <p>Set the property to true to improve the runtime performance.</p> <p>Use NOLOCK for SELECT statements to avoid locks on the database or table when SELECT statements are issued. An example syntax is:</p> <pre>select * from dbo.D_MailerIndex with (nolock) where ...</pre>
<code>be.backingstore.optimize.writes</code>	
	<p>Used with Microsoft SQL Server only.</p> <p>Set the property to true to improve the runtime performance.</p> <p>Use ROWLOCK with UPDATE or DELETE statements to avoid lock contentions. When you use ROWLOCK in the T-SQL statement, the SQL Server locks only the rows that match the 'where' condition and not the entire table. An example syntax is:</p> <pre>DELETE FROM dbo.D_Mailed WITH (ROWLOCK) where mailernumber = '12345678895' and time_created\$ = 'somedate'</pre>
<code>be.backingstore.timestamp.useDateTimeZone</code>	
	<p>Used when the backing store is enabled.</p> <p>Set the property to true to ensure that the correct DateTime properties are retrieved when an agent's time zone is changed and the agent is restarted.</p>
<code>be.engine.cluster.cleanup</code>	
	<p>Used with write-behind strategy only. By default, deleted entities are removed from the backing store automatically at system startup. Set this property to false to disable that behavior.</p> <p>Default is true.</p>
<code>be.engine.cluster.recovery.threads</code>	
	<p>Recovery threads are used when pre-loading the cache during startup.</p> <p>For an explanation of pre-loading and other pre-loading controls, see At system startup, domain object settings will determine how entity objects are stored and pre-loaded from the backing store to cache..</p> <p>Default is 5.</p>
<code>be.engine.cluster.recovery.distributed.strategy</code>	

Property	Notes
	<p>This property is used for Shared All and for Shared Nothing persistence.</p> <ul style="list-style-type: none"> For Shared All: Possible values for this property are <code>batch</code> and <code>nobatch</code>. Batch mode is a distributed batch mode, where one cache node is the 'director' and gives jobs to other nodes while recovering data from the backing store. Therefore, more than one node is needed. All nodes need to be started at once so jobs gets distributed evenly. While in <code>nobatch</code> mode, each node tries to pick up a job by itself while recovering data from the backing store. Default is <code>nobatch</code>. <div data-bbox="491 583 534 625"></div> <div data-bbox="603 562 1471 655">By using the JMX Mbeans > Pre-load and Recovery Information, you can view which seeders are performing recovery depending on the strategy mentioned in CDD as either Batch and NoBatch in the JConsole.</div> <ul style="list-style-type: none"> For Shared nothing: The <code>be.engine.cluster.recovery.distributed.strategy</code> parameter has the five following recovery policies added as part of the recovery options. When shared nothing persistence is implemented and recovery is issued, then the policy determines when and how recovery can be made. The following are the allowed values for the five recovery policies: <ul style="list-style-type: none"> The <code>no_data</code> recovers the space without any data. This is same as removing shared nothing persistence files. The <code>data_loss</code> recovers the space with available data from each seeder. If recovery is done with missing seeders, there is a potential for data loss, because not all members are started, to ensure that all data is recovered. This policy ensures best-effort recovery with the available data. The <code>no_data_loss</code> recovers the space only if there are enough members to be able to recover the data before shutdown. If enough seeders are not available to recover the previous state of the cluster, recovery throws an exception. The <code>fast_load_only</code> recovers the space only if all of the members that were active before the shutdown is available in the cluster. This policy enables fast recovery. If there are more or less cluster members than before the shutdown, recovery throws an exception. When the <code>fast_load_only</code> is used; it is advisable to set Cache Agent Quorum to the total number of cache nodes. This increases the chances of a successful recovery. The <code>robust_load_only</code> forces slow recovery. <p>Default value is: <code>no_data_loss</code>.</p> <div data-bbox="451 1543 494 1585"></div> <div data-bbox="563 1486 1439 1648">The seeder information (the current seeder list) is stored in the Shared Nothing persistence files during shutdown. This information is then used during startup or restart to perform recovery. If the cluster is exactly the same during startup (that is, exactly the same members are available and quorum is satisfied), then fast loading of the data is performed.</div>
<code>be.engine.cluster.recovery.distributed.batchsize</code>	

Property	Notes
	<p>When distributed batch recovery is enabled (<code>be.engine.cluster.recovery.distributed.strategy=batch</code>), the recovery manager divides the target table into many smaller batches and assigns 1-to-n number of these batches to each node (for them to execute recovery).</p> <p>This parameter provides an approximation to the size of each such batch. Since batches are defined using the target table's key field (for example, approximated using ID\$ column), actual batch size will differ depending on the key distribution.</p> <p>Users can instead define <code>be.engine.cluster.recovery.distributed.batchpernode=2</code> and prevent that too few, or too many batches are created.</p>
<code>be.engine.cluster.useDBBatching</code>	
	<p>Note</p> <p>For use with cache aside and only when the parallel operations feature is used.</p> <p>This property has no effect if <code>Agent.AgentClassName.dbOpsBatchSize</code> is set to 1 (see Table 14, CDD Agent Classes Tab Inference Agent and Query Agent Properties,).</p> <p>This property affects how all RTC transactions that a database writer thread takes from the database operations queue are written to the backing store:</p> <ul style="list-style-type: none"> • When set to true, the RTC transactions are handled as one job. • When set to false, each RTC transaction is handled as a separate job. <p>For a guide to usage of this and other related properties, refer to the "Database Write Tuning Options for Cache Aside" section in the <i>TIBCO BusinessEvents Architect's Guide</i>.</p> <p>Default value is false.</p>
<code>be.jdbc.multibyte.support</code>	
	<p>You can specify whether multibyte characters are supported in the column values.</p> <p>Set the value to true for the Microsoft SQL Server where multibyte character support is required.</p> <p>The default value is false.</p>

Setting Up Shared Nothing Persistence

Shared Nothing persistence allows you to store data at individual node level, instead of a centralized location.

Following steps explain how to configure a TIBCO BusinessEvents project to use Shared Nothing persistence:

Procedure

1. In BusinessEvents Studio Explorer, edit the project CDD file to set the Object Management to Cache and Provider to TIBCO.



Shared nothing persistence works in the cache only mode. Any overrides for the concepts or events should also be in the cache only mode.

2. For the Backing Store, select the Persistence Option as `Shared Nothing`.
3. Enter the value for persistence path as the absolute path to the directory where data is to be stored.
4. Select the appropriate persistence policy.
5. Save the CDD file.

Result

When using up Shared Nothing persistence, to ensure that data is not lost when nodes leave the cluster, you must set the number of backup copies to 1 or more.



The following words are reserved words and cannot be the name of a property in an event or a concept: `extid`, `id`, `closure`, `next`, `ttl`, and `fired`.

Runtime Configuration to Specify the Engine Name Property

The TIBCO BusinessEvents engine name is used to name the Shared Nothing files for that node.

To enable the engine name to be used, you need to explicitly set the engine name property when running clusters with Shared Nothing.

The `-n <name>` must be specified for all the nodes (inference and cache) in the cluster.

Note that only one node will be started in the following cases:

- If the cache nodes in your cluster do not specify any names, the hostname of the machine on which the engine is running is used as the default engine name.
- If the cluster contains cloned cache nodes (nodes with the same engine name), only one cloned node will start.

Nodes with duplicate engine names will not be initialized.

Recovery Options for Shared Nothing Persistence

You can use five policies for the shared nothing persistence as recovery options.

The `be.engine.cluster.recovery.distributed.strategy` parameter is also supported for the shared nothing persistence. When shared nothing persistence is implemented and recovery is issued, then the policy determines how and when the recovery can be made. The default value of the property is `no_data_loss`.

Policy	Policy Description
<code>no_data</code>	Recovers the space without any data. This is same as removing shared nothing persistence files.
<code>data_loss</code>	Recovers the space with available data from each seeder. If recovery is done with missing seeders, there is a potential for data loss, because not all members are started, to ensure that all data is recovered. This policy ensures best-effort recovery with the available data.
<code>no_data_loss</code>	(Default). Recovers the space only if there are enough members available to recover the data but fast-batch mode replication is not possible. to be able to recover the previous state of the cluster. Otherwise recovery throws an exception.

Policy	Policy Description
<code>fast_load_only</code>	Recovers the space only if the <code>no_data_loss</code> conditions are met and replica entries can be distributed among cluster members in fast-batch mode, When the <code>fast_load_only</code> is used; it is advisable to set 'Cache Agent Quorum' to the total number of cache nodes. This increases the chances of a successful recovery.
<code>robust_load_only</code>	Recover the space only if there are enough members to be able to recover the data before shutdown. If enough seeders are not available to recover the previous state of the cluster, recovery throws an exception. This policy forces the slow recovery of the space.
<code>force_load</code>	Forces recovery of the space even if the old shared nothing persister files are renamed as per new setup and hostname. This policy bypasses required host, seeder checks and loads the data anyway to complete recovery from old shared nothing files.

The `be.engine.cluster.as.minSeeders` property's value is the Cache Agent Quorum value minus the number of backup copies. The seeder information (the current seeder list) is stored in the shared nothing persistence files during shutdown. This information is then used during startup or restart to perform recovery. If cluster is exactly the same during startup (that is, exactly the same members are available and quorum is satisfied), then fast loading of the data is performed.

Berkeley DB Shared All Persistence

Berkeley DB Shared All Persistence uses TIBCO BusinessEvents DataGrid as the cache provider and Oracle Berkeley DB Java Edition software as the data store (not bundled with the TIBCO BusinessEvents software).



You must obtain a separate license for the Oracle Berkeley DB Java Edition software that is appropriate for your usage if you wish to use the Shared All with Berkeley DB feature.

With this form of persistence, cache servers act as persisters, and interact with the persistence layer. All cache nodes must have access to a reliable shared file system. The Berkeley DB data store is created within this shared file system.

The Berkeley DB Shared All Persistence option uses stores on a shared file system, such as NFS. This can result in better performance than use of a traditional DBMS product accessed over the network.

Reading from the Data Store

Persisters load data in bulk from the data store at startup. The following functions are used to load individual concepts into cache:

```
CacheLoadConceptByExtIdByURI()
CacheLoadConceptById()
```

Writing to the Data Store

Data is flushed to the operating system buffers for every write operation. TIBCO BusinessEvents uses `WRITE_NO_SYNC` durability by default, which means that Berkeley DB will flush every write to the operating system's buffers immediately but not call `fsync`. In case of an application crash, there will be no data loss as long as the underlying operating system synchronizes its buffers to disk.

Notes



- TIBCO BusinessEvents internally sets durability to `WRITE_NO_SYNC` durability. This behavior can be overridden by setting the `je.txn.durability` property in the `je.properties` file. See [Configuring the Berkeley DB Shared All Persistence Option](#).
- The `CacheLoadConceptByExtId()` function is not supported for use with this feature.
- Scorecards are not persisted (and so cannot be recovered).
- Scheduler events are not persisted (and so cannot be recovered)

Configuring the Berkeley DB Shared All Persistence Option

You can configure Berkeley DB Shared All Persistence in the CDD file and optionally in a properties file called the `je.properties` file.

Procedure

1. Download the supported version of the Oracle Berkeley DB Java Edition software from the Oracle web site. See the Release Notes for details on supported versions. Place the `je-versionNumber` JAR file in the following directory:
`BE_HOME/hotfix/lib/ext/tpcl/`
2. Open the project CDD or add one if the project does not yet have a CDD.
 - If you add a new CDD select Cache object management type in the wizard.
 - If you edit an existing CDD, select the Cluster tab. If the Object Management node is set to In Memory, right-click Object Management and select Cache.

3. Select the Cluster tab. In the navigation tree, select Object Management. In the Configuration panel, set the following:
 - Provider: **TIBCO**
 - Cache Agent Quorum: Generally set to the number of cache agents (see [CDD Cluster Tab Cache OM Settings](#) for details)
 - The following items are not relevant: Number of Backup Copies, Entity Cache Size, Object Table Cache Size.
4. In the Cluster tab navigation tree, select **Backing Store**. In the Configuration panel set the following values:
 - Persistence Option: **Shared All**
 - Database Type: **Berkeley DB**
 - Berkeley DB Data Store Path: The name and file location of the data store and (if used) the `je.properties` file.
5. If the CDD is using overrides for concepts or events, select the check box "Has Backing Store" to indicate that the entity will be stored in Berkeley DB. If the check box is not selected, then the entity will not be stored.
6. In the Cluster Tab navigation tree, select **Properties**. In the configuration tab, add the following properties:
 - `be.backingstore.useobjecttable` (see [CDD Cluster Tab Backing Store Properties Reference](#) for details).
 - `be.engine.cluster.as.lock.ttl` (as the property is deprecated, use the **Lock Timeout** option in [CDD Cluster Tab Cache OM Settings](#)).
7. In the Agent Classes tab, configure inference agents with the `be.engine.cluster.as.node.retry.times` property (see [CDD Cluster Tab DataGrid Properties Reference](#) for details).
8. Save the CDD file and build project EAR files for deployment.
9. If you will use any properties shown in [Reference To Berkeley DB \(JE\) Properties](#), add the properties as name-value pairs in a file called `je.properties`. Place the file in the same location as the data store (as specified in the Berkeley DB Data Store Path setting).

Reference to Berkeley DB Shared All Persistence CDD Properties

This section provide a reference to the CDD property.

Berkeley DB Shared All Persistence CDD Configuration Properties

Property	Notes
Cluster Settings	
Berkeley DB Data Store Path	<p>Specifies the name and file location of the data store. If this setting is not specified, a directory named <code>datastore</code> is created under the working directory.</p> <p>If you use a <code>je.properties</code>, file place it in the same location as the data store. See Reference To Berkeley DB (JE) Properties .</p>



As the `be.engine.cluster.as.lock.ttl` property is deprecated, use the 'Lock Timeout' option in [CDD Cluster Tab Cache OM Settings](#).

Reference To Berkeley DB (JE) Properties

This section provides a reference to the Berkeley DB JE properties you can use to override the default configuration.

For more details on the Berkeley DB JE properties, refer to the Berkeley DB product documentation:

- <http://www.oracle.com/technetwork/database/berkeleydb/je-faq-096044.html>
- http://docs.oracle.com/cd/E17277_02/html/GettingStartedGuide/administration.html#propertyfile

The properties are added to a file called `je.properties`. See [Configuring the Berkeley DB Shared All Persistence Option](#) for the required location.

Berkeley DB JE Properties

Property	Notes
<code>je.txn.durability</code>	
	Indicates the durability of a transaction. Possible values are <code>WRITE_NO_SYNC</code> , <code>NO_SYNC</code> , and <code>SYNC</code> . TIBCO BusinessEvents internally sets durability to <code>WRITE_NO_SYNC</code> . You can override that setting using this JE property.
<code>je.log.numBuffers</code>	
	The number of write buffers that the Berkeley DB uses internally. Default value is 3.
<code>je.log.bufferSize</code>	
	The size of each write buffer. Default value is 1048576.
<code>je.log.totalBufferBytes</code>	
	The sum of sizes of all the write buffers. Default value is 3145728.
<code>je.maxMemoryPercent</code>	
	The percentage of the JVM maximum memory to which the cache is limited. Default value is 60%.

Domain Objects Configuration

At system startup, domain object settings will determine how entity objects are stored and pre-loaded from the backing store to cache.



If a backing store is already set up, and you enable any Use Backing Store settings, you must update the backing store setup. The backing store will not operate correctly unless you do so. See [Updating Existing Backing Store Schema](#).

Domain object settings let you configure various behaviors for objects generated by the inference engines and stored in a cache. Many options relate to the way objects move between cache and backing store, so that you can tune memory usage and performance as needed.

You can configure the various behaviors globally (at the default level) and you can set overrides at the object type level. (Not all object level settings, however, are overrides.)

The main options are as follows. (Other options pertain to more specific situations and all are documented in the reference tables.)

- The mode: Cache Only, or Memory Only. See Cache Modes and Project Design in *TIBCO BusinessEvents Architect's Guide* to understand the effect of the different modes.
- Whether the objects or handles or both are preloaded into the backing store at startup.
- A preload fetch size (one setting for both objects and handles).
- Whether the cache is limited or unlimited. If limited, you can specify the size of the cache. See [Configuring a Limited \(or Unlimited\) Cache](#).

At the individual object type level only, you can also configure the following:

- Whether the object is stored in the backing store or not.
- A backing store table name (used when setting up a backing store. See [JDBC Backing Store](#)).

The settings are applied at the object level. For example, a contained concept can have a different limited cache setting from its container concept, and could be evicted from the cache at a different time.

Preloading Options

Preloading refers to the loading of the cache with objects from the backing store, at system startup.

If you do not preload, then the objects are loaded as needed, which can affect performance the first time each object is requested, depending on the size of the objects.

At the default level, you can choose to preload or not preload two types of items separately: the objects themselves, and the handles to the objects, which are stored in a separate table called the object table. You can also specify the number of objects to preload (one setting for both types of items).

At the individual object level, you can override the preload setting as follows:

- Use the default setting
- Preload (True)
- Do not preload (False)

You can also specify (or override) how many objects to preload.



One tuning property for preloading is available: `be.engine.cluster.recovery.threads`

Configuring Preloading Options

You can configure the preloading options for domain objects.

Procedure

1. Add a CDD, or open an existing CDD, configured for Cache OM.
2. Select the Cluster tab > Domain Objects > Default node on the left and on the right, configure settings as explained in [CDD Cluster Tab Domain Objects Default Settings Reference](#).
3. To add object-level overrides and other object-level settings do the following:
 - a) Select the Cluster tab > Domain Objects > Overrides node and click **Add**.
 - b) At the Entity Selection dialog, select the ontology object type or types you want to configure and click **OK**. The first entity in the list is selected and the configuration section for the entity displays on the right.
 - c) Select the `/uri` node for each selected entity in turn and configure the settings on the right as needed. You can also edit existing override entries, and remove entries not needed (by clicking Remove). See [CDD Cluster Tab Domain Object Override Settings Reference](#).



When a project is migrated from 4.x to 5.1, domain object override entries are automatically added for all entities. See Migration from 4.x to 5.1 in TIBCO BusinessEvents Installation for details.

4. Save the resource.

CDD Cluster Tab Domain Objects Default Settings Reference

The Domain Object Default settings apply to all objects except those for which you explicitly configure overrides, using the Domain Object Overrides section.


See [CDD Cluster Tab Domain Object Override Settings Reference](#).

CDD Cluster Tab Domain Object Default Settings

Property	Description
Mode	<p>With Cache OM, you can keep memory objects in the cache or Rete network using the following cache modes.</p> <p>Memory Only Objects are not persisted in the cache. They are kept in the Rete network (working memory) only.</p> <p>Cache Only Objects are persisted in the cache. They must be loaded into working memory as needed. This is the most common choice for a cache cluster.</p> <p>Note If you set the mode to Memory Only, the rest of the properties in this section are not relevant and are ignored.</p> <p>See Cache Modes and Project Design in <i>TIBCO BusinessEvents Architect's Guide</i> to understand the effect of this setting.</p> <p>Default is Cache Only.</p>

Property	Description
Preload Entities	<p>Specifies whether objects are loaded into the cache from the backing store on system startup (both normal startup and recovery).</p> <p>Checked</p> <p>All objects are preloaded into the cache from the backing store. Lower level settings can override this setting by excluding specified objects.</p> <p>Unchecked</p> <p>No objects are preloaded. into the cache from the backing store. Lower level settings can override this setting by including specified objects.</p> <p>Default is unchecked .</p>
Preload Handles	<p>Specifies whether object handles are loaded into the <code>ObjectTable</code> cache. The <code>ObjectTable</code> cache holds references (handles) to the objects themselves.</p> <p>Handles are used in the object table.</p> <p>Checked</p> <p>All object handles are preloaded. Lower level settings can override this setting by excluding handles for specified objects.</p> <p>Unchecked</p> <p>No object handles are preloaded into the cache from the backing store. Lower level settings can override this setting by including handles for specified objects.</p> <p>Default is unchecked .</p>
Preload Fetch Size	<p>If Preload Entities or Preload Handles or both are checked, this setting specifies the number of entity objects or handles (or both) to preload for each entity type whose objects or handles (or both) are configured to be preloaded.</p> <p>This setting applies to both objects and handles and cannot be set differently for each. Objects and handles are fetched in a non-deterministic manner.</p> <p>This setting can be overridden at the entity level.</p> <p>Set to 0 to preload all. Set to a number to load that number of objects or handles (or both).</p> <p>Default is 0. Ignored unless Preload Entities or Preload Handles or both are checked.</p>

Property	Description
Check for Version	<p>This field applies to concepts that use cache-only mode.</p> <p>An inference agent uses its L1 cache, a local cache of limited size, to improve access time to the concepts stored in the cluster cache. When an agent finds a concept instance in this local cache, the Check for Version setting determines whether the agent will use the instance directly, or instead check in the cluster cache for more recent version.</p> <p>If Checked (default value) The agent checks in the cluster cache for a more recent version. If a more recent version exists, it will be used, and will replace the one found in the local cache.</p> <p>If Not Checked The agent uses the instance found locally.</p> <p>When content-aware load balancing is used, the local instance can be used without checking for version, improving performance.</p> <p>Default is checked .</p>
Constant	<p>This field applies to entities that use the cache-only mode. In multi-engine deployments, use only for entities that do not change after creation.</p> <p>The processing unit has a special local cache used only for entities marked as Constant. Entities placed in this cache are only removed when they are explicitly deleted. If the processing unit finds an entity in the constant cache, it will use it without checking in the cluster.</p> <p>If Checked The entity is marked Constant, and uses the constant cache.</p> <p>If Not Checked The entity does not use the constant cache.</p> <p>Default is unchecked .</p>
Evict from Cache on Update	<p>Used only if both of the following are the case:</p> <ul style="list-style-type: none"> • Cache-aside database write strategy is used • The property <code>Agent.AgentClassName.cacheTxn.updateCache</code> is set to false (see CDD Agent Classes Tab Properties Reference) <p>If checked: When a rule action changes the value of any of an entity's properties or inserts a new entity, then the entity instance is evicted from the cache (updates are saved in the backing store)</p> <p>Use as needed to improve performance and cache memory management. For example, if an entity is not accessed frequently, it may save memory in the cache if the entity is evicted from cache after it is updated.</p> <p>Possible values are checked (true) and unchecked (false).</p> <p>Default is unchecked .</p>

Property	Description
Is Cache Limited	<p>If checked, the cache size is limited.</p> <p>Limited cache requires use of a backing store. See Configuring a Limited (or Unlimited) Cache.</p> <p>The size of the entity cache and the size of the object table cache are set in the Object Management section of the Cluster tab.</p> <p>If not checked, the cache size is unlimited.</p> <p>You can override this default setting in entity overrides.</p> <p>Default is unchecked.</p>
Concept TTL	<p>Time-to-live (in seconds) for the concept. After the time-to-live is expired, ActiveSpaces expires the concept within acceptable limits of the timeout.</p> <p>The default value is -1, which means concept does not expire, and must be explicitly consumed.</p> <p>Note: Following are some constraint for using the Concept TTL field with BusinessEvents:</p> <ul style="list-style-type: none"> • Containment relationship: Ensure that parent and child concepts expire together or that the time-to-live for the parent is less than time-to-live for the child concept. Thus, the application never references the child once the parent has expired. • Reference Relationships: Do not set this field for concepts that reference other concepts, or for concepts that are referenced in other concepts. <div>  <p>After the time-to-live expires, the concept is removed from the object table after a specified delay interval. This delay helps reduce object table load. You can configure this interval using the <code>objectTable.eviction.delay.interval</code> CDD property. Thus,</p> <p><i>Actual eviction time from object table = TTL + objectTable.eviction.delay.interval</i></p> <p>The default value is 60 seconds.</p> </div>

CDD Cluster Tab Domain Object Override Settings Reference

Many settings simply override the value of default settings.

See [CDD Cluster Tab Domain Object Default Settings](#) for general details about the use of each setting. The table below only provides details that are specific to overrides.




CDD Cluster Tab Domain Object Override Settings

Property	Notes
Entity URI	
	Specifies the project path to the entity for which overrides are being set. Defaults to the selected entity's URI. For example: <code>/Concepts/MyConcept</code> .
Mode	

Property	Notes
	<p>Overrides the Default level setting for this object type.</p> <p>Memory Only Mode</p> <p>If you set the mode for an entity to Memory Only, the rest of the properties in this section are not relevant and are ignored. Backing store is disabled for entities that use Memory Only mode.</p> <p>Caution</p> <p>If you change from Memory Only mode to a cache mode after the backing store has been set up, you must update the backing store schema. See Updating Existing Backing Store Schema .</p>
Preload Entities	
	<p>Specifies whether objects of the specified type are loaded into the cache from the backing store on system startup (both normal startup and recovery).</p> <p>Overrides the Preload Entities setting at the Default level.</p> <p>default</p> <p>Use the Preload Entities setting specified at the default level.</p> <p>true</p> <p>Objects of the specified type are preloaded into the cache from the backing store. If the default level setting is not to preload entities, you can use this override to preload selected entities.</p> <p>false</p> <p>No objects of the specified type are preloaded into the cache from the backing store. If the default level setting is to preload entities, you can use this override to not preload selected entities.</p> <p>Default is <code>default</code> .</p>
Preload Handles	
	<p>Specifies whether object handles for the specified type are loaded into the cache from the backing store on system startup (both normal startup and recovery).</p> <p>Overrides the Preload Handles setting at the Default level.</p> <p>default</p> <p>Use the Preload Handles setting specified at the default level.</p> <p>true</p> <p>Handles for the specified type are preloaded into the cache from the backing store. If the default level setting is not to preload handles, you can use this override to preload selected entities' handles.</p> <p>false</p> <p>No handles for the specified type are preloaded into the cache from the backing store. If the default level setting is to preload handles, you can use this override to prevent preloading the selected entities' handles.</p> <p>Default is <code>default</code> .</p>

Property	Notes
Preload Fetch Size	
	Overrides the Preload Fetch Size setting in the Default settings.
Check for Version	
	Overrides the value of the same-named setting in the Default settings.
Constant	
	Overrides the value of the same-named setting in the Default settings.
Evict from Cache on Update	
	Overrides the value of the same-named setting in the Default settings.
Is Cache Limited	
	Overrides the value of the same-named setting in the Default settings.
Concept TTL	
	Overrides the value of the same-named setting in the Default settings.
Backing Store Section	
Has Backing Store	
	<p>Used only if the Backing Store > Enabled checkbox is checked. To exclude an entity from the backing store, uncheck the Has Backing Store checkbox.</p> <p>Caution</p> <p>If you enable this override setting after the backing store has been set up, you must update the backing store schema. See Updating Existing Backing Store Schema .</p> <p>Concepts Related by Containment or Inheritance</p> <p>All concepts related by containment or inheritance must either be included in the backing store or excluded from the backing store. That is, they must share the same value for the Has Backing Store setting.</p> <p>Default value is checked .</p>
Table Name	
	Specifies a table name to be used in the backing store. Typically used if the entity name is long. See Ontology Identifiers that Exceed the DBMS Maximum Column Length for details.
Properties Metadata Section	
Property	

Property	Notes
	Displays the property name. Read-only.
Present in Index	
	<p>If checked, sets an unordered index on the property.</p> <p>Used only if Oracle Coherence is the cache provider. Also used only if the query agent has enabled indexing: see <code>be.agent.query.enable.filter.optimizer</code> in CDD Agent Classes Tab Properties Reference.</p>
Encrypted	
	<p>Specifies whether the property should be encrypted. The values are:</p> <ul style="list-style-type: none"> • true • false (default) <p>Note:</p> <ul style="list-style-type: none"> • For field level encryption to work, cluster level "Security" must be enabled and policy file must have 'data_encryption=true' set in it. • Fields that are indexed cannot be selected for encryption. • Fields that are used in query filters should not be encrypted.
Max Length	
	<p>Used with backing store to specify the length of string properties that exceed 255 characters (that is the actual contents stored in the column is more than 255 characters). Specifies the expected maximum length for the property. See String Properties that Exceed the DBMS Maximum Column Length for details.</p>
Reverse References	

Property	Notes
	<p>This setting is for use only with ConceptReference type concept properties.</p> <p>With a backing store, database updates related to a referring concept in a referenced concept can cause decreased performance. This happens when there are very many reverse references in a shared instance (referenced by many other instances).</p> <p>To address this issue, set the value to <code>false</code> for ConceptReference type properties.</p> <p>If you set the value to <code>false</code>, you must explicitly remove ConceptReference properties for deleted referenced concepts in the referring concept in your code.</p> <p>For example, if <code>employee</code> is a ConceptReference type property in a concept <code>acme</code>, and <code>smith</code> is an instance of a concept type <code>employee</code>, then you would set <code>ReverseReferences</code> to <code>true</code> for the <code>employee</code> ConceptReference property, and you would add something like this to rules:</p> <pre>acme.employee = null; Instance.deleteInstance(smith);</pre> <p>Or, for array properties:</p> <pre>Instance.PropertyArray.removeConceptReference(acme.employee, smith); Instance.deleteInstance(smith);</pre> <p>Default is <code>true</code>.</p>
Composite Indexes Section <p>The table lists the composite indexes for the entity. Composite indexes are based on multiple columns (entity properties). You can add, delete, and edit a composite index from the list.</p> <p>Click the Add icon () to create a composite index. In the Create Composite Index window, select the properties that you want to include in the composite index and click OK. A new composite index (with an autogenerated name) is displayed on the composite indexes list.</p>	
Index Name	<p>Displays the name of the composite index.</p> <p>You can perform the following actions on the selected composite index:</p> <ul style="list-style-type: none"> • Edit the composite index - Select the index row and click the Edit () icon to edit the composite index. You can either add a property or remove a property from the index. • Delete the composite index - Select the index row and click the Delete () icon to delete the composite index.
Properties	<p>Displays the properties that are included in the composite index.</p>

Collections, Agent Classes, and Processing Units

You can perform advanced configurations for destinations in the Agent Classes tab. (Destinations that are added to agent classes individually can be configured at the Agent Classes tab.)

Collections

At the Collections tab, you can (optionally) group rules, rule functions, and destinations into collections so that they can be easily assigned to agent classes (and processing units in the case of log configurations).

See [Configuring Collections of Rules, Rule Functions, and Destinations](#).

Agent Classes

Agent classes define the different sorts of agents you can deploy. Various agent types are available depending on the object management (OM) type and on the add-on products used. Each agent type is configured differently.



See [Using Properties at Different Levels](#) to understand the effect of using agent class properties at the cluster level and at the processing unit level to widen the scope of the property.

Configuring Agents with Collections and Individual Resources

Different agent types use different types of resources.

In the Agent Classes node (on the left side of the CDD editor) you see categories of collections. Here, you add collections you defined earlier, as needed to configure the agent class.

Rules

(Inference agent classes only.) It can be convenient to organize rules into collections for use in different inference agent classes. Select rule collections and individual rules as needed to define what rules will execute on inference agents of such classes at runtime.

Input Destinations

Different agents listen for messages arriving at different destinations. When you select a destination for use in a collection or an individual agent, you add deploy time configuration settings, to create a *destination configuration*. For example, you define an event preprocessor and a threading model to use. Each destination configuration is assigned a unique ID.

In the configurations for Input Destinations under **Collections**, you can optionally specify this rule function under **Thread Affinity Rule Function**.



This is only available for **Shared Queue** and **Destination Queue** threading configurations. When specified, the return value from the rule function is used to pick up the thread to which the message will be dispatched. The rule function is called with the event as the parameter, and is expected to return a `not null` value. Events that return the same value, will always be assigned to the same thread.

It is required that the rule function handles its exceptions and does null checks and returns `non-null` values. Failure to do so will cause the message allocation to a thread to fail also. In other words, the message will fail to propagate.

The rule function should be lightweight and should only perform read-only operations on the event or its payload.



It is only a convenience mechanism to identify, compute and return the relationship key. It should not perform the wider range of operations that are allowed in rule functions used elsewhere such as in pre-processors or in rules, such as `load`, `create`, `update`, `delete` objects, acquire locks and so on. Doing so may cause unknown behavior.

Startup Functions and Shutdown Functions

Select function collections and individual functions as needed, to define which functions execute at engine startup and shutdown respectively.

How Startup Rule Functions and Shutdown Rule Functions are Executed

- The order of the functions (including the order of functions within collections) is the order in which they execute at runtime.
- Put startup rule functions (for use at start up) into different collections from shutdown rule functions (those used at shut down) so you can select them appropriately at the agent classes tab.

Log Configurations

Also in the Collections tab, you can add different log configurations. These are used when you configure processing units.

See [Log Configurations](#).

Configuring Collections of Rules, Rule Functions, and Destinations

The purpose of configuring collections of rules, rule functions, and destinations is to make it simpler to configure agent classes. When you configure an agent, you can add collections of resources or individual resources or both. Two collections are predefined: an all-rules collection and an all-functions collection.

The procedure is in general the same for rules, rule functions, and destinations, so in these instructions, the word *item* is used to refer to any rule or rule function or destination.

A collection can have references to items (rules, rule functions, or destinations), and also references to other collections of the same type. References are identified in the groups tree by a reference symbol (↗). This mechanism enables you to reuse collections for more efficient configuration.

For the log configurations procedure, see [Configuring Log Configurations](#)

Procedure

1. In the Collections tab do any of the following:

- To add a new collection, select the parent for the collection type, Rules, Destinations, Functions, or Log Configurations as needed, and click **Add**.
- In the Item Collection field that appears on the right, enter a name for the group and click **Add** again.

Ensure that the collection name is unique across different collections in the CDD. For example, Rule collections and Destination collections in the CDD cannot have the same name.

- To add *items* and *item* group references to a collection, select the item collection and then click **Add**.

You see the Select Items dialog box.

2. In the Select *Items* dialog box do any of the following:

- To add *items*, in the **Items** tree click the check boxes of *items* you want to add to the group you are defining.
- To add collection references, in the **Collection References** tree click the check boxes of collections you want to add (by reference) to the collection you are defining.

When you select a collection on the left, you see details on the right: For example, the path to item you selected, and the names of collections you selected.

3. For function collections only, reorder the functions as needed, so that they execute in the correct order at runtime (that is, at startup or shutdown). Highlight a rule function in the tree on the left, and then click Move Up or Move Down as needed.
4. For destination collections only, configure each destination in turn. Select the destination on the left and complete the settings on the right to define characteristics such as the threading model to use, and the event preprocessor. See [CDD Collections Tab Input Destination Settings Reference](#) for information about each setting.
5. Save the CDD.

Updating Collections

You can update the already configured collections in several ways.

Procedure

1. To remove an item in a collection or the collection itself, select the item or the group on the left and click **Remove**.
2. To reorder rule functions in a function collection, select a rule function in the tree on the left, then click **Move Up** or **Move Down**. This is important for startup and shutdown rule functions. Ensure that you put startup and shutdown rule functions into appropriate separate collections.
3. You can change the URI (project path) of project resources to match their actual locations. To change the URI of an item, select the item on the left and edit the URI on the right.

CDD Collections Tab Input Destination Settings Reference

These agent-specific destination configuration settings are also available from the Agent Classes Tab. Collections enable you to configure once and use in multiple agents..

CDD Collections Tab Input Destination Settings

Property	Notes
Input Destination ID	Uniquely identifies this destination configuration at runtime. Edit as needed to ensure that each destination in the cluster has a unique deployment name. Default value is destination name.
URI	Project path to the destination (that is path to the destination in the design-time project).
Preprocessor	Specifying an event preprocessor for a destination is optional. Tip If you specify a preprocessor, you generally also specify worker thread settings, because event preprocessors are multithreaded (unless Caller's Thread threading model is used, which is single-threaded). Select the rule function that has been configured as this destination's event preprocessor. Event preprocessors are rule functions with one argument of type simple event.

Property	Notes
Threading Model	<p>If you specified a preprocessor, also specify thread settings. Select one model:</p> <p>Shared Queue</p> <p>Uses the TIBCO BusinessEvents system-wide shared queue and threads. For queue size and number of threads settings, see CDD Agent Classes Tab Settings Reference.</p> <p>Caller's Thread</p> <p>Uses the thread (and queue size) provided by the channel resource client. There is one thread per destination.</p> <p>Note</p> <p>If it is important to ensure that acknowledgements are sent in the expected order with Caller's Thread threading model, do not use the parallel operations option. See <code>Agent.agentClassName.enableParallelOps</code> in CDD Agent Classes Tab Inference Agent and Query Agent Properties.</p> <p>Destination Queue</p> <p>TIBCO BusinessEvents creates a dedicated thread pool and set of worker threads in each destination. See Thread Count and Queue Size below.</p> <p>For more information on threading models see Threading Models and Tuning in <i>TIBCO BusinessEvents Architect's Guide</i>.</p>
Thread Count	If you specified Destination Queue in the Threading Model setting, specify the number of threads for this destination here.
Queue Size	If you specified Destination Queue in the Threading Model setting, specify the queue size for this destination here.

Agent Classes (All OM Types)

Agent class types are as follows:

Cache Agent

Used only with Cache OM. Cache agents handle distributed cache object storage. A processing unit can contain only one cache agent.

LiveView Agent

Used only with Cache OM. LiveView agents provides configurations required for TIBCO BusinessEvents and TIBCO Live DataMart integration. by using the LiveView agent you can connect the TIBCO BusinessEvents project to the TIBCO Live Datamart server.

Query Agent

Used only with Cache OM and available only if TIBCO BusinessEvents Event Stream Processing software is used.

Dashboard Agent

Used only with Cache OM and available only if TIBCO BusinessEvents Views software is used. See *TIBCO BusinessEvents Views Developer's Guide* for details.

Monitoring & Management Agent

Used internally by the Monitoring and Management component. Do not add any agents of this class. Configuration of this agent type is explained in the "Basic MM Configuration" section of *TIBCO BusinessEvents Administration*.

Process Agent

Used only with Cache OM and available only if TIBCO BusinessEvents Process Orchestration software is used.

Inference Agent

Used with all OM types. For inference agent classes, you distribute a project's resources among the agent classes to define the specific work each agent will do. In Memory OM uses only inference agents, and each agent operates independently. With Cache OM, the agents share the cache data (as explained in TIBCO BusinessEvents Architect's Guide).

Adding an Agent Class

You can begin by configuring classes provided by the wizard. You can rename the classes as desired. Then add more classes as needed.

Procedure

1. In the Agent Classes tab, click **Add Agent**.
2. In the New Agent Class dialog box enter an Agent Class Name and select an Agent Class Type from the list (see section introduction). Valid types for your project depend on object management type, and whether you use any TIBCO BusinessEvents add-on products. Click **OK**.
3. The new agent class name appears on the left. Select the agent class name. Appropriate settings for that agent type appear in the Configuration section.

- Complete the settings as explained in [CDD Agent Classes Tab Settings Reference](#)
- Add any properties as needed. The available properties are explained in [CDD Agent Classes Tab Properties Reference](#).

4. For inference, dashboard, and query agent class types, add the resources you want to use. In the agent class tree on the left, click each type of resource collection in turn and configure as explained next.

(In the instructions below, the word *item* stands in for destination, function, and rule depending on the collection category.)

- a) Highlight a category of collections (for example Input Destination Collections).
- b) Click **Add**. You see the Select *items* dialog box.
- c) In the upper section of the dialog box, select individual project *item* resources, as desired.
- d) In the lower section of the dialog box (the Reference Groups section), select *item* collections you defined earlier, as desired.
- e) Click **OK**. A list of *item* IDs appears in the box on the right.

5. If you added any individual destinations to the Input Destinations Collections category, highlight their name on the left and configure their settings on the right. See [CDD Collections Tab Input Destination Settings Reference](#) for details.

(Destinations within input destination collections are configured at the Collections tab.)

6. Do any of the following as needed:
 - Click a collection category on the left to see a list of collections and *items* you selected from that category on the right.
 - Expand a category on the left and click a collection reference within it. You see a list of its item IDs and paths, and any collection references within that collection, on the right.
 - Edit the project paths for individual items you add here. You would do this only if the project location of that item changed.

7. Save the CDD.

Adding a LiveView Agent to the BusinessEvents Project

By using the LiveView agent you can connect the TIBCO BusinessEvents project to the TIBCO Live Datamart server.

The LiveView agent is added to a processing unit similar to how other agents are added to the processing unit. The LiveView agent joins the cluster as leech and only to those spaces which are configured to publish data to the Live DataMart server.



Ensure that in the CDD file, under the **Cluster** tab the **Store Properties As Individual Fields** check box is selected for the **Object Management: [Cache]** configuration. If this check box is not selected then the values of all the fields in the Live DataMart table are null.

Procedure

1. In the BusinessEvents Studio, open the project CDD file for editing.
2. In the **Agent Classes** tab, click **Add Agent**.
3. Provide a **Agent Class Name** and select LiveView as **Agent Class Type**. Click **OK**.
The new LiveView agent is listed under the Agent Classes. The LiveView agent has the **LDM Connection Configuration** and **Entity Filter Configurations** listed under it.
4. Under the Agent Classes list, select the **LiveView** agent and provide values for the configurations.
See [CDD Agent Classes Tab Settings Reference](#) for more details on the configurations.
5. Select **LDM Connection Configuration** and provide values for the following configurations.

Configuration	Description
LDM Url	Specify URL of the TIBCO Live Datamart server.
User Name	User name for the TIBCO Live Datamart server.
Password	Password for the User Name .
Initial Size	Specify the initial size of the connection pool.
Max Size	Specify the maximum size of the connection pool.

6. Select **Entity Filter Configurations** and provide values for the following configurations.

Configuration	Description
Entity Filter Configurations	Specify entities which you want to display in the LiveView dashboard. You can also specify a filter to select only those instances which qualifies the filter. For more information on adding a new filter and its configurations, see Adding Entity Filter Configurations for the LiveView Agent on page 78.
Generate Liveview Files	Select the check box to generate the LiveView configuration (.lvconf) files at the location specified in the Output Directory .

Configuration	Description
Output Directory	Specify the folder where the LiveView configuration (.lvconf) files are stored after generation. You can provide this location while starting the Live Datamart server to load the project data directly into the server.

- In the Processing Units tab, select the processing unit under which you want to add the LiveView agent. Under the **Agents** section, click the **Add** icon and select the newly created LiveView agent and click **OK**.



You must add the LiveView agent in a different processing unit than the inference agent. BusinessEvents TransformerFactory might get mixed up with LiveView client API when both the agents: LiveView agent and inference agent, are in the same processing unit. In such case, an error is published in the BusinessEvents logs. This is a harmless error as no data is missed-out in LiveView. To avoid this error, place the LiveView agent and the inference agent in different processing units.


- Save the project.

Adding Entity Filter Configurations for the LiveView Agent

Specify entities which you want to display in the LiveView dashboard. You can specify a filter to select only those instances which qualifies the filter. Also, you can specify a trimming rule to trim the LiveView table based on the rule.

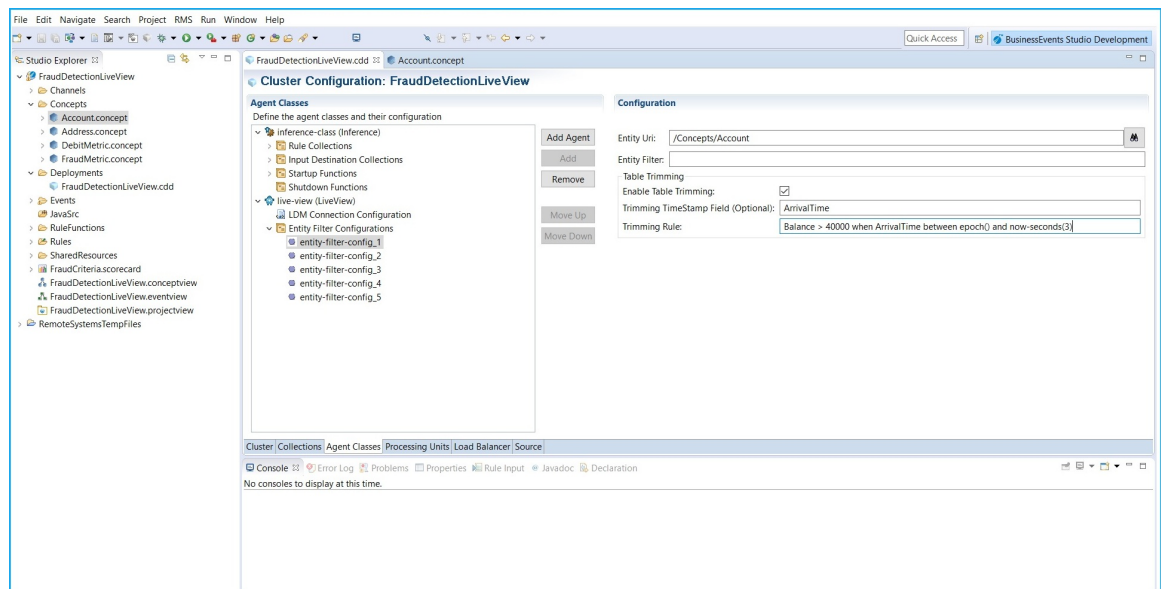
Procedure

- In the BusinessEvents Studio, open the project CDD file for editing and select **Agent Classes** tab.
- Under the Agent Classes list, select **Entity Filter Configurations** under an existing LiveView agent.
- Click **Add** to add an entity filter configuration.
- Specify the value for the following fields for the new filter and save the agent.

Field	Description
Entity Uri	Browse and select the supported entity that you want to send to the publisher based on your filter.  The supported entities are Concept, Metrics, and Events.
Entity Filter	Specify a query to filter out the entity based on your requirement. For example, the following values specifies to send the Account concept to the publisher only when Balance is greater than 10000. <ul style="list-style-type: none"> Entity Uri - /Concepts/Account Entity Filter - Balance > 10000 If this field is empty, then all instances of the entity are pushed to the Live Datamart server.

Field	Description
Enable Table Trimming	<p>Select the check box to trim or clean up the LiveView table automatically based on the user-defined rule specified in Trimming Rule. If the check box is selected, the following fields are displayed:</p> <ul style="list-style-type: none"> • Trimming TimeStamp Field (Optional) • Trimming Rule
Trimming TimeStamp Field (Optional)	<p>Specify name for an optional timestamp field for your entity.</p> <p>If your entity does not have a timestamp field to write the trimming rule, you can use this field to create a new timestamp field for the entity. This new field is part of the LiveView table after generating the LiveView configuration file.</p> <p>This field is displayed only when the Enable Table Trimming check box is selected.</p>
Trimming Rule	<p>Specify a rule to omit the entity data from the LiveView table that matches the trimming rule. The rule can use any field within the entity specified in the Entity Uri field. If required, you can also use Trimming TimeStamp Field for creating rules.</p> <p>For example,</p> <pre>OrdStatus=='BAD' when ArrivalTime between epoch() and now()-seconds(6)</pre> <p>where,</p> <ul style="list-style-type: none"> • OrdStatus is a property of an Account concept. • ArrivalTime is a property defined in the Trimming TimeStamp Field field. <p>This field is displayed only when the Enable Table Trimming check box is selected.</p>

Entity Filter Configurations for the LiveView Agent



CDD Agent Classes Tab Settings Reference

The following tables explain settings used with inference agents, query agents, and LiveView agents. Query agents are used with the TIBCO BusinessEvents Event Stream Processing add-on. Dashboard agents are used with the TIBCO BusinessEvents Views add-on and are documented in *TIBCO BusinessEvents Views Developer's Guide*.

CDD Agent Classes Tab Inference Agent and Query Agent Settings

Setting	Notes
Inference Agent and Query Agent Settings	
Max Size (Local Cache)	<p>Specifies the maximum number of objects (entities) in each agent's L1Cache (inference agent) or local cache (query agent). The L1 cache is a local cache used by the inference agent for local access to recently used objects. It is used to optimize access to objects.</p> <p>The query local cache is used in a way similar to the inference agent L1Cache. The query agent's local cache stores cache data locally for efficient reuse. The local cache listens to and synchronizes the locally stored entity instances with those in the main cache, so that the local cache stays up-to-date.</p> <p>When the threshold is reached, oldest entities are removed first.</p> <p>Default is 1024 (unit is objects).</p>
Eviction Time (Local Cache)	<p>Specifies an age limit on the cached entities in seconds. After this period, they are removed from the local cache.</p> <p>Note</p> <p>Age resets each time an entity is accessed by a query engine.</p> <p>Default is 900 .</p>
Queue Size (Shared Queue)	<p>Used for destinations whose threading model is Shared Queue (see Threading Model in CDD Collections Tab Input Destination Settings).</p> <p>Specifies the queue size for the processing unit-wide shared queue</p> <p>Note</p> <p>In this release, set the same value for all agents configured to deploy in the same processing unit.</p> <p>If set to 0 (zero), the queue size is unlimited.</p> <p>Default is 1024 .</p>

Setting	Notes
Thread Count (Shared Queue)	<p>Used for destinations whose threading model is Shared Queue (see Threading Model in CDD Collections Tab Input Destination Settings).</p> <p>Specifies the number of processing unit-wide shared threads.</p> <p>Note</p> <p>In this release, set the same value for all agents configured to deploy in the same processing unit.</p> <p>As a guideline, set the value to the number of processors available to the JVM.</p> <p>In MM Console, this thread appears with the name <code>\$default.be.mt\$</code>.</p> <p>Default value is 10 .</p>
Max Active	<p>Specifies the maximum number of active agents of this class. This value is used for fault tolerance. Deployed agents that are acting as standby can take over from active instances that fail. In many cases, there is no need to keep standby instances.</p> <p>A value of 0 indicates an unlimited number of active instances.</p> <p>For example, the Max Active field is set to 1 and you start two BusinessEvents LiveView engine. As per the Max Active field, only one engine is active while the other one is passive. If the active engine fails, the passive engine takes over and become active.</p> <p>See Fault Tolerance of Agents in <i>TIBCO BusinessEvents Architect's Guide</i> for more details.</p> <p>Default is 0.</p>
Inference Agent Settings	
BusinessWorks Repo URL	<p>If this project will integrate with a TIBCO ActiveMatrix BusinessWorks project, enter the Repo URL for the ActiveMatrix BusinessWorks project repo URL here.</p> <p>Use forward slashes.</p>
Concurrent RTC	<p>If checked, enables concurrent run to completion cycles, generally shortened to RTC cycles. (Also known in prior releases as concurrent Rete and <code>concurrentwm</code>).</p> <p>Concurrent RTC does not require cache OM but does require local locking.</p> <p>The number of concurrent cycles is determined by the number of available threads.</p> <p>See Collections Agent Classes and Processing Units for details. Also see Concurrency and Project Design in <i>TIBCO BusinessEvents Architect's Guide</i> for important information on using concurrency features.</p>

Setting	Notes
Check for Duplicates	By default, TIBCO BusinessEvents checks if the external IDs (@extId) of entities are unique within the current agent. If you want to check for uniqueness of external IDs across the cluster, check this check box. Performing this check affects performance. Default is unchecked.

LiveView Agent Settings


Configuration	Description
Name	Name of the LiveView agent.
Queue Size	Specify the size of the queue to limit the amount of data you want to accept while publishing to the LiveView tables. The Queue Size and Thread Count fields are used for scaling purposes depending on the data size and load.
Thread Count	Specify the number of threads for starting publishing the data to the LiveView tables. Some entities are larger than others, and thus in such cases, you might need multiple thread for processing them.
Properties	Add properties and specify the values as required.

CDD Agent Classes Tab Properties Reference

Properties are available for inference agents, cache agents, and query agents.

CDD Agent Classes Tab Inference Agent and Query Agent Properties

Setting	Notes
Inference Agent and Query Agent Properties	
<code>com.tibco.cep.runtime.channel.payload.validation</code>	XML event payloads are validated when this property is set to true. There may be some loss of performance due to the extra processing. Default is false.
<code>com.tibco.cep.runtime.threadpool.shutdown.timeout.seconds</code>	Specifies time (in seconds) to wait for the worker thread to complete before shutdown.
Inference Agent Properties	

Setting	Notes
<code>Agent.AgentClassName.recoveryPageSize</code>	<p>Specifies the number of entries per page to be used while recovering objects from the cache.</p> <p>For example, if you set the value to 10,000, then the engine loads handles in blocks of 10,000, instead of trying to load them in a single batch. Smaller batch sizes result in slower recovery. Experiment with batch size to establish the best batch size to use for your environment.</p> <p>A value of 0 means that the objects are recovered in one iteration.</p> <p>Default is 0.</p>
<code>Agent.AgentClassName.cacheTxn.updateCache</code>	<p>Used only if cache-aside database write strategy is used.</p> <p>If set to false: When a rule action changes the value of an entity's properties or adds a new entity, then the entity instance is evicted from the cache instead of updating it. Updates are saved in the backing store. Use this setting and <code>Agent.AgentClassName.threadcount</code> as needed to improve performance and cache memory management.</p> <p>This property interacts with the Cluster > Domain Objects setting, Evict From Cache on Update (and its override settings if any):</p> <ul style="list-style-type: none"> When this CDD property is set to true, the domain objects Evict From Cache on Update setting is ignored, in the agent for which the property is set. When this CDD property is set to false, the domain objects Evict From Cache on Update setting overrides this CDD property. <p>See CDD Cluster Tab Domain Objects Default Settings Reference for details on the Evict From Cache on Update setting.</p> <p>Possible values are <code>true</code> and <code>false</code>.</p> <p>Default is <code>true</code>.</p> <div data-bbox="794 1459 837 1501">  </div> <div data-bbox="906 1438 1497 1537"> <p><i>AgentName</i> in the property refers to the name of the agent which actually evicts entities, for example, <i>inference-class</i>.</p> </div>
<code>Agent.AgentClassName.threadcount</code>	<p>For use with cache aside and only when parallel operations feature is used (see <code>Agent.agentClassName.enableParallelOps</code>).</p> <p>Defines the number of <code>\$CacheWriter</code> threads performing cache writing jobs.</p> <p>See Threading Models and Tuning in TIBCO BusinessEvents Architect's Guide for usage guidelines.</p> <p>Default value is 2.</p>

Setting	Notes
<code>Agent.AgentClassName.checkDuplica tes</code>	<p>Affects how TIBCO BusinessEvents checks uniqueness of entity external IDs (@extId).</p> <p>If set to false, checks for uniqueness of external IDs within the agent</p> <p>If set to true, checks for uniqueness of external IDs across the cluster. Performing this check affects performance so use it with care.</p> <p>Default is false.</p>
<p>Inference Agent Database Writer Thread Tuning Properties</p> <p>Note</p> <p>For use with cache aside and only when the parallel operations feature is used.</p> <p>For a guide to usage, refer to the "Database Write Tuning Options for Cache Aside" section in the <i>TIBCO BusinessEvents Architect's Guide</i>.</p>	
<code>Agent.agentClassName.enableParall elOps</code>	<p>If true, parallel operations are used</p> <p>Post-RTC phase operations are done in parallel:</p> <ul style="list-style-type: none"> • Writes to the cache • Writes to the database (relevant only cache aside strategy is used) • Executes the actions list, for example, sends messages (events) and acknowledges events, as needed. <p>Use of parallel operations generally requires use of locking to ensure data integrity.</p> <p>If false, sequential operations are used</p> <p>All post-RTC phase operations are done on a single thread in the order shown above.</p> <p>This property is set to false for specific needs such as when Caller's Thread threading option is used.</p> <p>Another reason to set the value to false is to ensure that the system waits to send a reply event confirming that some work has been done, until the result of the work can be seen in the cache.</p> <p>Defaults to true only if cache-aside write strategy and concurrent RTC are both used. Otherwise defaults to false.</p>

Setting	Notes
<code>Agent.AgentClassName.dbthreadcount</code>	<p>Defines the number of database write threads available to process the RTC transactions from the queue, that is, the number of threads performing database writing jobs (\$DBWriter thread pool). Writes include applying entity inserts, updates, and deletes to the database.</p> <p>Although the limit is seldom reached, you can guarantee that a connection is always available for a dbwriter thread as follows. Set this field to the same value as CDD Cluster tab > Backing Store > Connection > Max Size field.</p> <p>Default is 2.</p>
<code>Agent.AgentClassName.dbOpsQueueSize</code>	<p>The size of the queue (a Java blocking queue) for database writing jobs.</p> <p>Zero (0) or a negative value means the queue size is unlimited.</p> <p>Note</p> <p>When the queue is full, all engine operations are blocked.</p> <p>Default is 8.</p>
<code>Agent.AgentClassName.dbOpsBatchSize</code>	<p>Used in the post-RTC phase. Sets the maximum number of RTC transactions that a database writer thread takes from the database operations queue and processes in one batch.</p> <p>Database write threads process the RTC transactions from the queue. The number of threads is defined by <code>dbthreadcount</code>.</p> <p>A database write thread takes up to the <code>dbOpsBatchSize</code> number of RTC transactions, processes them and commits them to the database. (When database write threads are idle, they take available jobs from the database operations queue, even if there are less jobs than <code>dbOpsBatchSize</code>.)</p> <p>Default is 10.</p>
Query Agent Properties	
<code>be.agent.query.localcache.prefetchaggressive</code>	<p>If set to true, then objects required for a query are prefetched while the query is executing.</p> <p>The prefetch feature improves performance, but CPU and memory usage increases as a result of the aggressive prefetching. You may have to try different values till you find the optimal settings for your environment.</p> <p>Ensure that the cache size is large enough to accommodate objects that are prefetched.</p> <p>Default is false.</p>

Setting	Notes
<code>be.agent.query.enable.filter.optimizer</code>	<p>Used only with Oracle Coherence as the cache provider.</p> <p>If set to true, the query agent attempts to use indexing that is enabled and defined.</p> <p>If set to false, indexes are ignored by this agent.</p> <p>See also Present in Index, in CDD Cluster Tab Domain Objects Default Settings Reference.</p> <p>Default is false.</p>
<code>be.engine.queryAgent.channel.disable</code>	<p>By default, query agents connect to channels. In some cases, however, query agents do not need to connect to channels. To prevent query agents from connecting to channels, set this property to true.</p> <p>The default value is false.</p>
Cache Agent Properties	
<code>be.engine.cacheServer.channel.disable</code>	<p>By default cache agents connect to channels. In most cases, however, cache agents do not need to connect to channels. To prevent cache agents from connecting to channels, set this property to true.</p> <p>The default value is false.</p>
<code>be.engine.cluster.scheduler</code>	<p>A single agent of the cluster acts as a scheduler. By default all agents of a cluster can act as a scheduler. To avoid that a Cache Agent acts as a scheduler add the property and set it to false for the Cache Agent classes.</p> <p>The default value is true</p>
LiveView Agent Properties	
<code>be.engine.ldm.publish.retry.wait</code>	<p>Specifies the time interval before retrying publishing data from BusinessEvents cluster to StreamBase server.</p> <p>The default value is 2000, that is 2 seconds.</p>
<code>be.engine.ldm.publish.max.retries</code>	<p>Specifies the number of retry attempts for publishing data from BusinessEvents cluster to StreamBase server.</p> <p>The default value is 5.</p>

Log Configurations

Each processing unit references a log configuration. The log configurations are defined in the Collections tab.

You can also replace the default line layout implementation with your own. You can also override the default logging mechanism in TIBCO BusinessEvents to use the log4j mode. See [Overriding The Default Logging Mode](#) for more details. For some custom log4j configuration examples, see [Custom Log4j Configuration Examples](#).

For more information about log4j, see [Apache wiki page](#).

Log File Settings

For a reference to the settings, see the [Files Section](#) in [CDD Collections Tab Log Configurations Settings](#).

- Log File Name and Location

Set the name and location of the log file for a log configuration using the Name and Directory settings. If you do not enter a leading slash, the files are stored relative to the working directory (the directory in which you start the `be-engine.exe` executable). If you do not specify a name, the engine name is used. If no engine name is specified the name defaults to `cep-engine.log`.

- Number and Size of Log files

You can also set the size of a single log file, the number of files to keep, and whether a log file is flushed when an engine starts, or whether entries are appended. [Log Configuration Levels and Syntax](#)

Log Configuration Levels and Syntax

In a log configuration that uses the provided line layout implementation, you select a level of logging for each *module* in the TIBCO BusinessEvents runtime.

Levels

A level corresponds to how much logging is filtered out. They are ordered where `all` is lowest and `off` is highest:

Level	Description
Off	Highest possible rank. Filters out all logging messages (turns logging off for the specified module).
Fatal	Logs only severe runtime errors that cause the application to stop running.
Error	Also logs runtime errors that might not cause the application to stop running.
Warn	Also logs potentially harmful runtime events or situations.
Info	Also logs runtime informational events of general interest.
Debug	Also logs detailed runtime informational events, for use in identifying issues.
Trace	Also logs even more detailed runtime information.
All	Lowest possible rank. Turns on all logging including any custom logging levels.

Syntax

Enabling a lower level automatically enables the higher levels. For example, enabling `info` automatically enables `fatal`, `error`, and `warning`.

Assign each module to a level using a space-delimited list. The levels are not case sensitive. The syntax is as follows:

```
module1:level module2:level ...
```

To assign a certain level of logging to *all* modules, use an asterisk:

```
*:info
```

This syntax means that logging for all modules is at the `info` logging level.

You can use the asterisk syntax and also specify exceptions that use a different logging level. For example:

```
*:info driver.tibrv:debug
```

This syntax means that all modules use logging level `info`, except the module `driver.tibrv` which uses `debug` level.

Configuring Log Configurations

See [Log Configurations](#) for an explanation of the logging levels, modules, and syntax details

Procedure

1. In the Collections tab select Log Configurations and click **Add**.
2. In the Configuration section, give the log configuration a name.
3. Add the log levels you want to enable in this configuration. See [CDD Collections Tab Log Configurations Settings Reference](#) for details on the fields.
4. To redirect the STDERR or STDOUT streams, check the **Enable** check box.
For details, see [Send to Terminal Section](#).
5. Save.
6. To specify a Custom Line Layout class, in the Collections tab select Log Configurations and click **Add**.
7. In the Configuration section, give the log configuration a name.
8. In the Custom Line Layout section, check the **Enable** check box, and complete the Class and Argument fields as shown in [CDD Collections Tab Log Configurations Settings](#).
9. Save.

Overriding The Default Logging Mode

You can disable the default logging mode and use the log4j mode for logging.

After the default logging mechanism is disabled, the log4j mode for logging is automatically activated, using the default log4j configuration located at `BE_HOME\lib\ext\tpcl\apache\log4j.xml`. You can also specify your custom log4j file location to be used for logging in the `be-engine.tra` file.

Procedure

1. In the **Collections** tab of the project's CDD, select an existing log configuration under the Log Configurations.
For details on how to add a log configuration, see [Configuring Log Configurations](#).
The log configuration properties are displayed.
2. Clear the **Enable** check box to disable the default logging mechanism.

Log Configurations Properties

3. Save the CDD file.
4. To use your custom log4j configuration, add the following property in the `be-engine.tra` file.
`java.property.log4j.configuration=[custom_log4j_path]`



If the file is not in the classpath but located elsewhere on your device, use the following path:

```
java.property.log4j.configuration=file:[custom_log4j_file_path]
```

Custom Log4j Configuration Examples

You can create your own custom log4j configurations that you can use for logging instead of the default `log4j.xml` file.

You can also see the official log4j wiki page for more information on log4j configurations at <http://wiki.apache.org/logging-log4j/Log4jXmlFormat>.

Following are the few example XML configurations for the some logging use cases, which you can use after overriding the default logging (see [Overriding the Default Logging Mode](#) for more information).

Multiple Log Files Using Single log4j Configuration

Define multiple `FileAppender` classed to create multiple logging as shown in the following example:

```
<appender name="FILE" class="org.apache.log4j.FileAppender">
  <param name="file" value="${be.home}/logs/sample.log" />
  <param name="append" value="false" /> <param name="threshold"
value="debug"/>
  <layout class="org.apache.log4j.PatternLayout">
    <param name="ConversionPattern" value="[%t] %d{HH:mm:ss,SSS} %-5p %l -
%m%n" />
  </layout>
</appender>
<appender name="FILE" class="org.apache.log4j.FileAppender">
  <param name="file" value="${be.home}/logs/sample2.log" />
  <param name="append" value="false" /> <param name="threshold"
value="info"/>
  <layout class="org.apache.log4j.PatternLayout">
    <param name="ConversionPattern" value="[%t] %d{HH:mm:ss,SSS} %-5p %l -
%m%n" />
  </layout>
</appender>
```

Rotating Log Files Based on Time

Define the `DailyRollingFileAppender` class to roll over log files based on time. The following example shows the configuration for rolling over the log file on midnight each day:

```
<appender name="FILE" class="org.apache.log4j.DailyRollingFileAppender">
  <param name="file" value="${be.home}/logs/sample.log" />
  <param name="DatePattern" value="'.'yyyy-MM-dd" />
  <layout class="org.apache.log4j.PatternLayout">
    <param name="ConversionPattern" value="[%t] %d{HH:mm:ss,SSS} %-5p %l -
    %m%n" />
  </layout>
</appender>
```

The following table displays the example values of the `DatePattern` parameter which defines the time when to roll over the logs.

DatePattern Values For DailyRollingFileAppender

Time	Value
Minutely	'.'yyyy-MM-dd-HH-mm
Hourly	'.'yyyy-MM-dd-HH
Half-daily	'.'yyyy-MM-dd-a
Daily	'.'yyyy-MM-dd
Weekly	'.'yyyy-ww
Monthly	'.'yyyy-MM

Real Time Streaming For Console Logs

Define the `FlumeAppender` class to allow applications to send events to the the console, as shown in the following example:

```
<appender name="flume" class="org.apache.flume.clients.log4jappender.Log4jAppender">
  <param name="Hostname" value="localhost" />
  <param name="Port" value="41414" />
  <param name="UnsafeMode" value="false" />
  <layout class="org.apache.log4j.PatternLayout">
    <param name="ConversionPattern" value="[%t] %d{HH:mm:ss,SSS} %-5p
    %l - %m%n" />
  </layout>
</appender>
```

CDD Collections Tab Log Configurations Settings Reference

You can define settings for log file in the **Collections** tab of the CDD.

CDD Collections Tab Log Configurations Settings

Property	Description
Name	Name of this log configuration.

Property	Description
Enable	Check the Enable check box to enable this log configuration. All other Enable settings are ignored if this check box is unchecked.
Levels	Space-separated list of levels and modules used in this log configuration. See Log Configurations for more details. Default is <code>info</code> .
Files	
Enable	Check the Enable check box to enable log files to be written. Configure the settings in this section to specify details. If this check box is unchecked, all other properties in this section are ignored.
Directory	Enter the absolute path to the directory in which you want to store the files. If you do not enter a leading slash, the files are stored relative to the working directory (the directory in which you start the <code>be-engine.exe</code> executable).
Name	Name of the log file. The default value is the engine name. If no engine name is set, then the default value is <code>cep-engine.log</code>
Max number	Number of log files to keep. When the Max size setting value is reached, a new log file is created for the next log entries. Files are created up to the Max number setting size. The oldest file is deleted when a new file is added after this value is reached. Default is 10.
Max size	Maximum size of one log file. Default is 10000000.
Append	If checked then new entries are added to the end of the file. If not checked, the contents of the file are flushed each time the engine starts.

CDD Collections Tab Log Configurations Settings

Property	Description
Send to Terminal	
Enable	Check the Enable check box to enable the redirection specified in this section. If this check box is unchecked, all other properties in this section are ignored.
Output redirection	If true, the STDOUT stream is written to the terminal. If <code>false</code> , it is not.
Error redirection	If true, the STDERR stream is written to the terminal. If <code>false</code> , it is not.
Custom Line Layout	

Property	Description
Enable	<p>Check the Enable check box to enable the custom line layout entries to take effect. Configure the settings in this section to specify details of a custom layout.</p> <p>If this check box is unchecked, all other properties in this section are ignored.</p> <p>If this check box is checked all properties in the other sections are ignored (except Name, and Enable in the upper section).</p>
Class	<p>The custom line layout class.</p> <p>This class must implement <code>org.apache.log4j.Layout</code> and must be available in the runtime classpath.</p> <p>The class needs 2 constructors:</p> <ul style="list-style-type: none"> • One with no argument • One with a single String argument, which receives the value of the Arguments field.
Arguments	<p>A String parameter used for the custom line layout class, if required:</p> <ul style="list-style-type: none"> • To use the constructor that requires an argument, specify the argument • To use the constructor that does not expect an argument, leave the field empty.



To override the default TIBCO BusinessEvents CDD logging properties, see [Disabling the Default Logging in CDD](#).

Additional log settings can be added to the **Processing Unit** tab **Properties**, see [CDD Processing Units Tab Settings Reference](#).

Logging for TIBCO BusinessEvents DataGrid

You must configure logging for TIBCO BusinessEvents DataGrid in TIBCO BusinessEvents separately. To do so, set the properties in the CDD file.

Properties in the CDD File

Property	Description
<code>be.engine.cluster.as.log.dir</code>	<p>The directory to which the TIBCO BusinessEvents DataGrid log files will be written. If unspecified, the logs will be written to the same directory as the TIBCO BusinessEvents logs.</p>
<code>be.engine.cluster.as.log.filename</code>	<p>File name of the log file.</p> <p>By default, the file name is <code><engineName>-as.log</code>.</p>

Property	Description
<code>be.engine.cluster.as.log.level</code>	The log level specifying how much logging is to be filtered out. See Configuration Levels for TIBCO BusinessEvents DataGrid Logging for details. By default, the log level is set to INFO.
<code>be.engine.cluster.as.logfile.count</code>	Specifies the number of rolling log files allowed. Count is specified in integer.
<code>be.engine.cluster.as.logfile.size</code>	Log files are rolled over to a new log file when the specified size limit is reached. Size is specified in bytes.
<code>be.engine.cluster.as.logfile.append</code>	Set to true/false, specifies whether to append the logs to new files.

Configuration Levels for TIBCO BusinessEvents DataGrid Logging

Following levels can be used to specify the level of logging for TIBCO BusinessEvents DataGrid logs. Note that the logging levels are case insensitive.

Level	Description
None	Highest possible rank. Filters out all logging messages (turns logging off for the specified module).
Fatal	Logs only severe runtime errors that cause the application to stop running.
Error	Also logs runtime errors that might not cause the application to stop running.
Warn	Also logs potentially harmful runtime events or situations.
Info	Also logs runtime informational events of general interest.
Fine	Also logs detailed runtime informational events, for use in identifying issues.
Finer	Also logs even more detailed runtime information.
Finest	Lowest possible rank. Turns on all logging including any custom logging levels.

Configuring the Date Format in the Log Files

As per your requirement and locale, you can customize the date and time format used in the BusinessEvents logs.

Procedure

- Add the `be.trace.date.format` property to the project's CDD file or `be-engine.tra` and set its value with the required date and time format.
The date and time formats conforms to the formats of the `java.text.SimpleDateFormat` class of JDK 1.6, or the `org.apache.commons.lang.time.FastDateFormat` class of Apache. Refer the URL <http://>

docs.oracle.com/javase/6/docs/api/java/text/SimpleDateFormat.html for more information on the supported formats.

For example, setting the value to "yyyy MMM dd HH:mm:ss:SSS z" translates to "2015 Feb 23 10:47:23:456 IST" in the logs.

```
be.trace.date.format = "yyyy MMM dd HH:mm:ss:SSS z"
```

Processing Units (All OM Types)

To configure a processing unit (PU), you add the items you configured earlier, and any additional properties required. If you do not find a configuration item you require, click the appropriate tab and add it, then return to Processing Units tab and continue configuration.

One processing unit named `default` is provided out of the box. You can change this name. It has no significance, except that TIBCO Administrator expects a processing unit of this name by default, which can be useful for testing purposes.

Processing units are referenced in the site topology file used by TIBCO BusinessEvents Monitoring and Management component (see Basic MM Configuration in *TIBCO BusinessEvents Administration*).

PUs with Unique Agent Instance Properties

Depending on configuration, some processing units can be deployed more than once in a cluster. Others have unique configuration details that make them deployment-specific, that is, that limit them to being deployed only once in a cluster. Configuring the following CDD settings for a processing unit makes it deployment-specific processing unit:

Agent Key

At the processing unit node, you can associate a unique key with a selected agent class. This key identifies an agent instance uniquely at runtime. The purpose of the agent key is to retrieve scorecards from the backing store. Scorecards are local to an agent and the key enables the correct scorecard to be returned to the correct agent.

Agent Priority

The agent priority determines which agents of a given class are active, when fault tolerance is used. Each deployed agent of an agent class can have a different priority; however, if the agent have the same priority then cluster decides which agents it has to activate.

See "Deployment-Specific Processing Units and Global Variables" section in *TIBCO BusinessEvents Administration* for other ways a processing unit can be deployment-specific.

Adding a Processing Unit

A processing unit deploys as a TIBCO Cloud Events engine. One engine runs in one JVM.

Each processing unit contains one or more agents of different types. The main types are inference agents which performs the inferencing work, and cache agents, which manage the objects. For details on the fields present on the Processing Units tab, see [CDD Processing Units Tab Settings Reference](#).

Procedure

1. In the CDD editor, click the **Processing Units** tab do the following:
 - Select the default processing unit and configure it. You can rename it as needed.
 - Click **Add** to add more processing units as needed.
2. In the **Name** field, enter the name for the processing unit as needed.

For deployment, TIBCO Administrator by default looks for a processing unit called `default` and a CDD file called `default`.

3. In the **Log Configuration** field, browse to and select one log configuration.
See [Log Configurations](#) for more details on log configurations.
4. Check the Hot Deploy check box if you want to enable hot deployment.
See *TIBCO BusinessEvents Administration* for details about hot deployment.
5. If you use the TIBCO BusinessEvents Data Modeling add-on, check the **Enable DB Concepts** check box to enable database concept functionality on this processing unit as desired.
6. In the Agents section, click **Add** and select an agent class.
7. If needed, assign to each agent a key and a priority.
See [PUs with Unique Agent Instance Properties](#) for details.
8. In the **Properties** section, add any additional configuration properties as required.
For example see the "Localhost and Localport Properties" section in [CDD Cluster Tab Coherence Properties Reference](#) for one use case.
9. Save the resource.

CDD Processing Units Tab Settings Reference

CDD Processing Units Tab Settings

Property	Notes
Name	Enter a name that is unique across the cluster.
Log Configuration	Browse to and select a log configuration, configured at Collections tab. See Log Configurations for more details.
Hot Deploy	Check the check box to enable hot deployment for this processing unit. See <i>TIBCO BusinessEvents Administration</i> for details about hot deployment.
Enable Cache Storage	<p>Check the check box to enable cache storage on this processing unit (PU). Settings available depend on the types of cache agents in the PU.</p> <ul style="list-style-type: none"> • In PUs used to host cache agents: The check box is checked and cannot be unchecked. • In PUs used only for dashboard agents (available in TIBCO BusinessEvents Views only): the check box is unchecked and cannot be checked. Dashboard agents cannot function as cache servers. • in PUs that host inference agents or query agents (or both): <ul style="list-style-type: none"> – If checked, the PU is used for storing cache data. – If unchecked, the PU is not used for storing cache data. <p>Note: Enable cache storage in PUs running inference and query agents for test deployments only. Not recommended in production. Default value for PUs containing inference agents or query agents (or both) is unchecked.</p>

Property	Notes
Enable DB Concepts	Check the check box to enable database concepts functionality for this processing unit. 'Available only with TIBCO BusinessEvents Data Modeling add-on software.
Agents Section	
Agent	Name of the agent class you selected. Agent classes are defined at the Agent Classes tab.
Key	Specifies a value that uniquely identifies an instance of an agent of this class at deploy time. Required for recovery of scorecards. Recommended in all cases, for situations that require an agent instance to be uniquely identified. The value for Key must uniquely identify the agent. Note: In certain TIBCO BusinessEvents Monitoring and Management methods, you may be prompted for a session name. For session name you generally put the agent class name. However, if the agent class also has a key, you must instead use the key value. Because of this, the key value must uniquely identify the agent. No default value.
Priority	Specifies the priority of the agent for load balancing purposes. The priority indicates the order in which standby agents become active, and conversely, the order in which active agents become standbys, when new agents join the cluster. The lower the number, the higher the agent is in the activation priority list. For example, an agent with priority 2 has a higher priority than an agent with a priority of 6. For agents with equal priority, the cluster decides which ones to activate. No default value.

CDD Processing Units Tab Coherence Log Properties Reference

The properties in the Coherence Log Properties section are used to configure the Coherence log. This log is used only if Oracle Coherence is used as the cache provider.

Standard logging settings are configured in the Log Configuration tab. See [CDD Collections Tab Log Configurations Settings Reference](#).

CDD Processing Units Tab Coherence Properties

Property	Notes
Coherence Log Properties	
<code>tangosol.coherence.log</code>	

Property	Notes
	<p>Specifies the output device used by the logging system.</p> <p>Optional.</p> <p>Possible values are:</p> <ul style="list-style-type: none"> • <code>stdout</code> • <code>stderr</code> • <code>jdk</code> (Requires JDK 1.4 or later) • <code>log4j</code> (Requires log4j libraries to be in the classpath) • A file path and file name <p>If you specify <code>jdk</code> or <code>log4j</code> you must also perform appropriate configuration of the JDK or Apache log4J logging libraries.</p> <p>Default is <code>stdout</code>.</p>
<code>tangosol.coherence.log.level</code>	
	<p>Specifies which logged messages are output to the log destination.</p> <p>Optional.</p> <p>Possible values are:</p> <ul style="list-style-type: none"> • 0: Only output without a logging severity level specified will be logged • 1: All the above plus errors • 2: All the above plus warnings • 3: All the above plus informational messages • 4-9: All the above plus internal debugging messages (the higher the number, the more the messages) • -1: No messages <p>Default is 5.</p>
<code>tangosol.coherence.log.limit</code>	

Property	Notes
	<p>Specifies the maximum number of characters that the logger daemon processes from the message queue before discarding all remaining messages in the queue.</p> <p>The message that causes the total number of characters to exceed the maximum is not truncated.</p> <p>All discarded messages are summarized by the logging system with a single log entry detailing the number of discarded messages and their total size. When the queue empties, the logger is reset and subsequent messages are again logged.</p> <p>The purpose of this setting is to avoid a situation where logging can itself prevent recovery from a failing condition, for example by contributing to timing issues.</p> <p>Logging occurs on a dedicated low-priority thread to further reduce its impact on the critical portions of the system.</p> <p>Optional.</p> <p>Possible values are positive integers or zero (0). Zero implies no limit.</p> <p>Default is 0</p>
JMX Management properties	
	You must specify the following JMX-related properties to enable cluster statistics to appear in the monitored objects table in MM.
<code>tangosol.coherence.management</code>	
	Default is all
<code>tangosol.coherence.management.remote</code>	
	Default is true

CDD Processing Units Tab JMS Server Connection Properties


You can add the JMS server connection properties at the cluster level if they apply to JMS channels in all processing units in the cluster.

See *TIBCO BusinessEvents Developer's Guide* for details about configuring a JMS channel.

CDD Processing Units Tab Properties for Reconnecting to a JMS Server

Property	Notes
<code>com.tibco.tibjms.connect.attempts</code>	

Property	Notes
	<p>Specifies the number of reconnection attempts, and the interval between each attempt to connect to the JMS server.</p> <p>The value must use the format: <i>attempts, retry interval</i>.</p> <p>For example: 10, 500 means 10 attempts, with a 500-millisecond interval between each retry attempt.</p> <p>This property is used only for channels that have a TIBCO Enterprise Message Service provider.</p> <p>Note</p> <p>Use either <code>be.jms.reconnect.timeout</code> or <code>com.tibco.tibjms.connect.attempts</code>. If you set both the properties, then <code>com.tibco.tibjms.connect.attempts</code> takes precedence.</p> <p>Default is 2, 500.</p> <p>Note: The property does not work for the JNDI connection.</p>
<code>be.jms.ignore.startup.error.channels</code>	
	<p>Ignores the startup errors for the specified JMS URIs. You can specify a comma-separated list of JMS channel URIs as the value of the property. The default value is an empty string to retain existing behavior. Set the value to an asterisk (*) to ignore startup errors for all JMS channels.</p> <p>Also, set the property <code>be.jms.reconnect.timeout=<value in seconds></code> to retry the connection to Enterprise Message Service servers.</p>
<code>be.jms.reconnect.timeout</code>	
	<p>Specifies the retry interval (in seconds) for reconnecting to the JMS server when the connection is broken.</p> <p>A value of zero (0) means do not retry. Any other value means keep retrying (with no limit to number of retries), and use the specified interval between each attempt.</p> <p>If you require incremental interval between the reconnect attempts then set the <code>be.jms.reconnect.timeout.incremental.enabled</code> property.</p> <p>Note</p> <p>Unacknowledged messages (Events) are resent to the TIBCO BusinessEvents engine, which may result in duplicate events.</p> <p>Note</p> <p>Use either <code>be.jms.reconnect.timeout</code> or <code>com.tibco.tibjms.connect.attempts</code>. If you set both the properties, then <code>com.tibco.tibjms.connect.attempts</code> takes precedence.</p> <p>Default is 0 (zero)</p>
<code>be.jms.reconnect.timeout.incremental.enabled</code>	


Property	Notes
	<p>Using this property the random and incremental delays between JMS reconnect attempts are available. Thus, the reconnect requests are spaced out and do not send too many connect requests at once to the JMS server.</p> <p>The intervals start with the value of <code>be.jms.reconnect.timeout</code> as the minimum value and keep on increasing with each attempt up to a certain maximum limit, beyond which the interval is always the same as the maximum limit.</p> <p>The default value is <code>false</code>.</p> <p>This property can only be used along with the <code>be.jms.reconnect.timeout</code> property.</p>
<code>be.jms.reconnect.msgCodes</code>	
	<p>Specifies a case-insensitive character pattern that matches all error messages or error codes that will cause a reconnect attempt.</p> <p>This property is used for JMS channels with providers other than TIBCO Enterprise Message Service.</p> <p>Default is <code>*</code> (that is, the wildcard matched by any characters.)</p>
<code>be.channel.tibjms.queue.disabled</code> <code>be.channel.tibjms.topic.disabled</code>	
	<p>By default, be-engine connects to all defined channels on startup, including those not mentioned in the CDD file. This is because such channels can be used as output channels. However this is not always desired.</p> <p>To disable queue or topic connections for specific JMS channels, add the following properties as appropriate. Enter the project path to the JMS channel as the individual value. Use commas or spaces as the delimiter. Use forward slashes in the project path. For example:</p> <pre>be.channel.tibjms.queue.disabled=/channels/1jmschannel, / channels/3jmschannel be.channel.tibjms.topic.disabled=/channels/2jmschannel, / channels/4jmschannel</pre>
<code>be.channel.jms.unified</code>	
	<p>By default, TIBCO BusinessEvents creates two connections to a JMS server, with the client IDs.</p> <p>Set this property value to <code>true</code> for all agents to create a single connection with the same client ID as specified in the channel resource properties, or in the JMS Connection shared resource, if used. In the unified mode only a single value is required.</p> <div>  <p>When the connection is configured using a JMS Connection shared resource, ensure that the topic and queue connection factories on the JMS Connection shared resource Advanced tab match each other. Also, when using TIBCO Enterprise Message Service, use <code>GenericConnectionFactory</code> for both.</p> </div>

Property	Notes
<code>be.channel.jms.disallow.dup.clientid</code>	<p>Specifies whether the duplicate client ID can be used in the engine. When enabled, the engine fails to start when a duplicate client ID is encountered, even if the duplicate client ID is in the second engine.</p> <p>If the <code>be.channel.jms.disallow.dup.clientid</code> property is enabled in the default mode (not the unified mode), the JMS channel needs two client IDs (whitespace separated) to startup successfully (one for the queue connection, and one for the topic connection). In the unified mode, only one client ID is required.</p>
<code>be.channel.jms.sender.session.pool.maxsize</code>	<p>Specifies the maximum pool size for the JMS sender session pool. Once set, it activates the JMS sender session pool for each JMS channel in the project. By default no sender session pool is created, and a single shared session is used by sending functions per channel. This property is applicable for the non-transacted mode only. The minimum allowed value of the property is 1.</p>
<code>be.channel.jms.session.recover.onAckFailure</code>	<p>Specifies whether to autorecover a session that failed after acknowledgment failure of unexpired messages.</p> <p>To stop session autorecovery for such failures, set the <code>be.channel.jms.session.recover.onAckFailure</code> property to <code>false</code>.</p> <p>The default value is <code>true</code>.</p>
<code>be.jms.error.endpoint.enable</code>	<p>Specifies whether to enable forwarding of messages to error queue or topic for all destinations.</p> <p>If this property is enabled, TIBCO BusinessEvents forwards any unsupported message from the original destination queue or topic to an error queue or topic. These unsupported messages are then cleared from the original destination queue. The original message is acknowledged by BusinessEvents and proper error message is thrown for the unsupported message.</p> <p>If this property is enabled then you might also want to define <code>be.jms.default.error.queue.name</code> and <code>be.jms.default.error.topic.name</code> properties.</p> <p>The default value is <code>false</code>.</p> <p>Note: The unsupported type messages are forwarded to the error queue or topic on the same JMS server, on which the original message arrived. Therefore, in case of an application interacting with multiple JMS servers, individual error queue or topic are required on each server.</p>
<code>be.jms.default.error.queue.name</code>	

Property	Notes
	Name of the error queue. The default value is <code>be.application.error.queue</code> .
<code>be.jms.default.error.topic.name</code>	
	Name of the error topic. The default value is <code>be.application.error.topic</code> .
<code>be.engine.channel.<DestinationURI>.deliverydelay</code>	
	Set the delayed delivery time (in milliseconds) for all messages using the JMS channel destination.

StreamBase Channel Connection Properties

You can add StreamBase channel connection properties at cluster level if they apply to the StreamBase channels in all processing units in the cluster.

Property	Notes
<code>com.tibco.sbchannel.reconnect.attempts</code>	<p>Specifies the number of reconnection attempts to connect to the StreamBase server.</p> <p>The default value is 10.</p> <div>  <p>To turn off reconnection completely, set the value to 0. To set the infinite reconnection attempts, set the value to any negative number.</p> </div>
<code>com.tibco.sbchannel.reconnect.interval</code>	<p>Specifies the time interval, in milliseconds, between each attempt to connect to the StreamBase server.</p> <p>The default value is 5000 or 5 seconds.</p>

JDBC Backing Store

A backing store enables persistent backup of the objects generated and modified at runtime. Use of a backing store enables recovery in the event of a system-wide failure.

For instructions on migrating from the legacy Oracle Types (Oracle only) backing store to the JDBC backing store, see the migration chapters in TIBCO BusinessEvents Installation

The backing store feature requires use of Cache object management. Before you add a backing store, develop your caching solution and test it. Also ensure that your project ontology is completely configured.

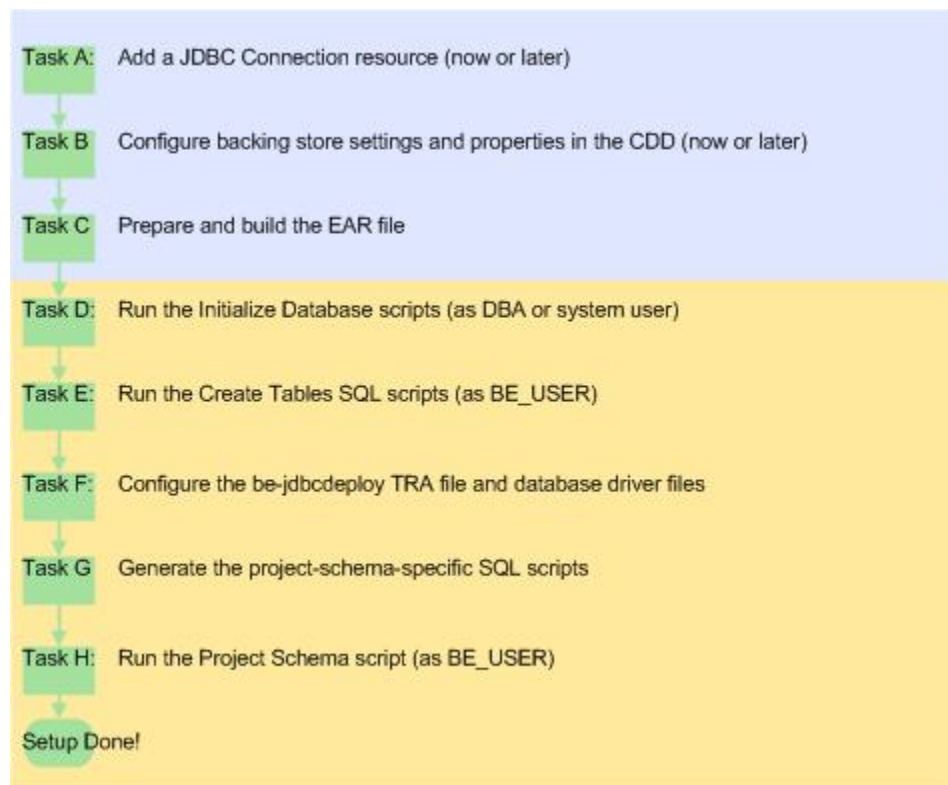


- Use a separate schema and schema owner for each project, even if different projects use the same ontology (otherwise ontology conflicts can occur).
- If your project ontology changes after the backing store is in place, you must update the backing store schema. See [Updating Existing Backing Store Schema](#).

The upper (blue) area in the diagrams shows TIBCO BusinessEvents Studio configuration tasks. The lower (yellow) area shows database setup utility tasks. The tasks shown map to task sections in this chapter.

For the simplest case where no additional project configuration is required (see [Ontology Identifiers that Exceed the DBMS Maximum Column Length](#)).

Main Tasks in Setting up a Backing Store



Task D is not available for DB2 since DB2 uses the OS runtime authentication system. Therefore, this step (run the initialize database script) does not apply to DB2.

Complete Task Flow



Task F is not available for DB2 since DB2 uses the OS runtime authentication system. Therefore, this step (run the initialize database script) does not apply to DB2.

For more information see [Ontology Identifiers that Exceed the DBMS Maximum Column Length](#).

Backing Store Setup and Configuration

Setup refers to using the provided scripts to create the backing store schema for your project.

See [Resources Required for Setting Up the Database](#) for the DBMS-related requirements.

For the basic setup tasks, see [Initializing the Database and Generate Non-Project Tables](#). However it is important that you first read the section [Ontology Identifiers that Exceed the DBMS Maximum Column Length](#) to understand if you need to do any special project configuration before running the scripts. (Other project configuration can be done before or after you do the setup tasks.)

In addition to setting up the backing store (and doing related project configuration for special cases), you also configure the TIBCO BusinessEvents Studio project to use the backing store according to your needs. You can do the following to configure backing store behavior:

- Use the Shared All persistence option, with Database Type Oracle, MySQL, Postgres, or SQL Server (formerly known as JDBC backing store).
- You can use either write-behind or cache-aside database write strategy.
- Tune the database connection pool properties.
- You can use a limited or unlimited cache. You can define a global setting and configure object level overrides.
- Control how the cache is preloaded from the backing store at startup. You can define a global setting and configure object level overrides.

See [Configuring Backing Store Settings in the CDD \(Now or Later\)](#).

You must also add a JDBC Connection resource to your project, before or after the backing store setup.

To make the flow of tasks simpler, all TIBCO BusinessEvents Studio project configuration is documented before the setup tasks, because some TIBCO BusinessEvents Studio configuration is required in certain cases. The procedures make it clear which configuration can be done before or after backing store setup.

Excluding Entities from the Backing Store

You do not have to use the backing store for all entities. In the CDD file you can specify entities for which you do not want to use the backing store.



If later you want to include any excluded entities, you must change the setting and update the backing store setup as explained in [Updating Existing Backing Store Schema](#).

See [Configuring CDD for Special Cases \(As Needed\)](#) for details.

Ontology Identifiers that Exceed the DBMS Maximum Column Length

Entity names and entity property names are used by backing store scripts to generate database table and column identifiers.

DBMSs put different limits on the length of a database identifier name. For example, in Oracle the maximum length is 30 characters, and in SQL Server the limit is 128 characters.

Generated database identifiers are longer than the TIBCO BusinessEvents identifiers because they contain characters in addition to the TIBCO BusinessEvents identifier. You can handle long identifier issues in either of the following ways (or a combination of these ways).

Letting the Utility Generate Short Aliases for Table Names

You can allow the `be-jdbcdeploy` utility to generate short aliases for long names. You can also edit those names and rerun the utility. For details see [Configuring Aliases File and Project Schema Script](#).

Note that alias file entries are also generated for another reason. See [Ontology Identifiers that Use Database Key Words](#).

Specifying Short Table Names in the CDD File

You can avoid the problem of long entity type names before you begin to configure the backing store by specifying short database identifiers using the CDD Table Name setting.

The advantage of this method is that you can choose meaningful names before running the `be-jdbcdeploy` utility. The disadvantages are that you may not know ahead of time which entities require short names, and you must also ensure that the table names you specify are unique across all entities in the ontology.

If you do not specify table names, and entity names are repeated, on the other hand, the generated table names are appended with dollar (\$) characters as necessary, for example, D_ORDER, D_ORDER\$, D_ORDER \$\$ and so on.

See [Configuring CDD for Special Cases \(As Needed\)](#) for details.

Ontology Identifiers that Use Database Key Words

As well as database names that are too long, ontology terms that are key (reserved) words in your DBMS product must also be mapped to an alias. If errors occur when you run the SQL scripts due to key word clashes, examine the errors and add the appropriate words to the key word mapping file.

A provided file (*BE_HOME/bin/dbkeywordmap.xml*) ships with some basic mappings: start, end, schema, mode, and index. You can use it as a model.

Unlike the Aliases file, the key word mapping file is not a project-specific file. It is intended to be generally useful across different projects. However, keyword mappings are also added to the aliases file when you run the SQL scripts, so you can also provide project-specific aliases for the generic mappings, if you want to.

The procedures are explained in the task sections within the section [Initializing the Database and Generate Non-Project Tables](#).

String Properties that Exceed the DBMS Maximum Column Length

The default column size for String type attributes is 255 characters. If you expect the data length of an entity property to exceed that value, then in the CDD file set the Max Length field for each entity's properties. The utility changes the data type of String attributes with long lengths to CLOB, as appropriate.

See [Configuring CDD for Special Cases \(As Needed\)](#) for details.

Resources Required for Setting Up the Database



For specific database products and versions supported, see the readme file, which is in the *TIBCO_HOME/release_notes* directory.

Provided Configuration Resources

The table below lists resources required and sections following explain the procedures for setting up backing store tables.

Resources Required for JDBC Backing Store Implementation

Resource	Default Location and Notes
Provided Files in <i>BE_HOME/bin</i>	
<code>base_types.xml</code>	The <code>base_types.xml</code> file is used by the deployment utility. Do not edit this file.
<code>be-jdbcdeploy</code> executable <code>be-jdbcdeploy.tra</code>	Only used for manual SQL script generation. Generally not needed. You can use a TIBCO BusinessEvents Studio option instead. (See Generated SQL Scripts below).

Resource	Default Location and Notes
<pre>create_tables_oracle.sql create_tables_sqlserver.sql create_tables_db2.sql create_tables_mysql.sql create_tables_postgres.sql</pre>	<p>Use the appropriate SQL (DDL) script for your DBMS. This script creates the tables that are used to maintain the metadata.</p> <p>The script drops any existing tables and recreates them.</p>
dbkeywordmap.xml	<p>This file contains mappings to handle words used in the TIBCO BusinessEvents project that are database reserved words. See Ontology Identifiers that Use Database Key Words for details.</p>
<pre>initialize_database_oracle.sql initialize_database_sqlserver.sql initialize_database_mysql.sql initialize_database_postgres.sql</pre>	<p>Use the appropriate script for your DBMS.</p> <p>By default the user is called BE_USER with the password BE_USER and the user has DBA rights. Edit the script if you want the user to have a different name or different rights.</p> <p>For SQL Server, this script also creates the default database, with the name BE_USER and makes it the default database for the user BE_USER.</p> <p>Note:</p> <p>Use a different user (and schema) for every TIBCO BusinessEvents project that needs a backing store. This script drops the user (and therefore all the tables) and adds the user again.</p>
<p>Generated SQL Scripts</p> <p>These scripts are generated when you use the File > Export > JDBC Deployment wizard. The value for <i>yourname</i> is specified in the Output Script Name Prefix setting. You specify the location of the scripts in the wizard</p> <p>You can manually execute the be-jdbcdeploy executable. You specify script name prefix at the command line. Scripts are generated in the same directory where you run be-jdbcdeploy.</p>	
yourname.sql	This SQL (DDL) script creates schema tables and types.
yourname.aliases	<p>This file has entries if the database table identifiers are longer than the DBMS maximum character limit. See and Configuring Aliases File and Project Schema Script.</p>

Resource	Default Location and Notes
<code>yourname_alter.sql</code>	The <code>yourname_alter.sql</code> script is for use in schema migration. Generated only after updates are made to the <code>be-jdbcdeploy.tra</code> file. See Updating Existing Backing Store Schema .
<code>yourname_cleanup.sql</code>	For use as needed. This script truncates the tables.
<code>yourname_delete.sql</code>	This script is used for deleting entities that have been marked as deleted (applies only if write-behind database write strategy is used).
<code>yourname_remove.sql</code>	For use as needed. This script removes the database schema. You can use it to reset the project.

Install Prerequisites for DBMS Software

You must install prerequisites for the DBMS software before you begin to configure the backing store.

Procedure

1. Install and start a supported DBMS product.
See the product readme file for a list of supported products.
2. Copy the appropriate JDBC drivers file to `BE_HOME/lib/ext/tpcl`.
3. Restart BusinessEvents Studio Explorer after copying the drivers file.
This step is required before you can use the design-time Test Connection feature. It is also required for runtime.
4. Now or later: If you will use the debugger or tester features, add your DBMS product's libraries to the TIBCO BusinessEvents Studio classpath.
The remainder of this section provides a few tips for each supported DBMS.

SQL Server

Here are a few helpful points about SQL Server:

SQL Server authentication for non-production purposes

It is convenient to use SQL Server authentication so you can create database users as needed. Select this option when you install Microsoft SQL Server. With Windows Authentication, on the other hand, you may have difficulties creating users without help from others in your enterprise.

Availability Group

An availability group must be ready and dedicated to TIBCO BusinessEvents so that the a TIBCO BusinessEvents database or TIBCO BusinessEvents databases can be added to that group.



It is required to create a full backup of the database before adding it the availability group.

SQL Server AlwaysOn Availability Groups

The AlwaysOn Availability Groups feature in SQL Server 2012 Enterprise Edition is a high-availability and disaster-recovery solution used by TIBCO BusinessEvents. To implement this feature, make sure that your SQL Server setup is up and running properly as per the Microsoft documentation.

Connection Properties to the SQL Server Cluster

Use the availability group listener defined at the SQL Server cluster level in the TIBCO BusinessEvents JDBC resource. The availability group listener enables clients to connect to a SQL Server replica without knowing the name of the physical instance of the SQL Server, for example:

```
jdbc:sqlserver://sqlserver-group-listener:1433;databaseName=be_user
```

To allow faster TCP connection retries, use the property `multiSubnetFailover=true` even if your cluster is on the same subnet as recommended by Microsoft,

Set this property in the BusinessEvents JDBC resource in the URL field, for example:

```
jdbc:sqlserver://sqlserver-group-listener:1433;databaseName=be_user;multiSubnetFailover=true
To support Windows authentication, use property integratedSecurity=true
```

See [SQL Server authentication vs. Windows authentication](#) for details.

Configuring Your Machine for Windows Authentication

This authentication is only supported on Microsoft Windows operating systems. In order for TIBCO BusinessEvents to support Windows authentication when accessing SQL Server database, follow these steps:

Procedure

1. Download and install the Microsoft JDBC 4 Driver distribution on the machines BusinessEvents runs on.
The distribution contains the `sqljdbc_auth.dll` used by the client to support Windows authentication.
2. Edit the `be-engine.tra` file and update the `tibco.env.PATH` variable so that it points to the folder where `sqljdbc_auth.dll` resides. For instance:
`tibco.env.PATH C:/sqljdbc_4.0/enu/auth/x64%PSP%%BE_HOME%/hotfix/bin%PSP%.`



JDBC Driver distribution has an x64 or x86 version of that dll. Pick the same version as your BusinessEvents installation.

3. Edit the `studio.ini` file and add the `java.library.path` variable so that it points to the folder where `sqljdbc_auth.dll` resides. For instance:
`-Djava.library.path=C:/sqljdbc_4.0/enu/auth/x64`



JDBC Driver distribution has an x64 or x86 version of that dll. Pick the same version as your BusinessEvents installation.

4. Use the property `integratedSecurity=true` in the BusinessEvents JDBC resource. For instance:
`jdbc:sqlserver://sqlserver-group-listener:1433;databaseName=be_user;integratedSecurity=true`
5. Replace the script in `<tibco_be_home>/bin/initialize_database_sqlserver.sql` with the following script that creates a user associated to Windows login:

```
use master
go
drop database be_user
go
create database be_user
go
```

```

drop login [domain\user]
go
create login [domain\user] from windows with default_database = be_user
go
use be_user
create user [domain\user] for login [domain\user]
go
grant control, alter, connect to [domain\user]
go
alter role [db_owner] add member [domain\user]
go

```

SQL Server authentication vs. Windows authentication

As mentioned in the Microsoft documentation, you should use Windows domain logins to access databases that are members of availability groups.

Database users that are associated to domain login added to a database in a primary replica are propagated to secondary databases, and then continue to be associated with the specified domain login.

In the other hand, database users associated with an SQL Server login will be propagated to the secondary databases without a login. The user will not be able to access data from any secondary database.

To overcome this behavior, give the SQL Server login a higher permission at the server level such as sysadmin on all the SQL Server replicas, or use Windows login.

For Windows login refer to [SQL Server AlwaysOn Availability Groups](#).

Datatype and Driver Information

The datetime datatype in SQL Server 2005 has the following range: 1/1/1753 to 12/31/9999. Microsoft SQL Server 2008 has added a new data type, datetime2, which has a date range of 0001/01/01 through 9999/12/31. Therefore, if you are using Microsoft SQL Server 2008, then you can manually change the generated SQL script (DDL) for your backing store, and replace any affected columns' data type from datetime to datetime2.

Use the SQL Server JDBC driver, sqljdbc4.jar. You can download this driver from: <http://msdn.microsoft.com/en-us/data/aa937724.aspx>.

Configuring Your Machine for Oracle Database

For configuring your machine for Oracle database, you need to install OCI drivers.

In production environments, you might have to ask a database administrator to create a database user for you. You should then be able to run the other SQL scripts yourself, logged on as the user created by the administrator.

Minimum User Permissions

By default the TIBCO BusinessEvents user permissions are set to DBA privileges. At a minimum, the user must be able to create tables and views. For example for an Oracle database you could use the following:

```

DROP USER BE_USER CASCADE;
CREATE USER BE_USER IDENTIFIED BY BE_USER;
GRANT CONNECT TO BE_USER;
GRANT RESOURCE TO BE_USER;
GRANT CREATE ANY VIEW TO BE_USER;
GRANT CREATE ANY TABLE TO BE_USER;

```

Installing an OCI Driver - OCI Driver Support

This procedure installs an OCI driver for the Oracle database.

Instructions assume you are working with a local database for testing purposes. Adapt the instructions if you are working with a remote database.

Procedure

1. Install an Oracle client: preferred as Admin/custom installation.
2. Admin/Custom installation will create the file `tnsnames.ora`.
Configuration while creating the file `tnsnames.ora` should match the database server configuration (database server is running on a remote machine).
3. Install TIBCO BusinessEvents and add the file `ojdbc7.jar` from the client installation to the folder `BE_Home/lib/ext/tpcl`.
4. The connect URL used in the TIBCO BusinessEvents is
`jdbc:oracle:oci:@DBServerHostName:1521:DBInstance`.
This URL can be found in the file `tnsnames.ora`.
5. If the installation was performed properly, the studio JDBC test connection should show success.

Oracle Real Application Cluster (RAC)

Oracle Real Application Cluster (RAC) supports both thin and OCI drivers.

The sample connection string for a thin driver is:

```
jdbc:oracle:thin:@(DESCRIPTION=(LOAD_BALANCE=on)
(ADDRESS=(PROTOCOL=TCP)(HOST=10.107.146.70)(PORT=1521))
(ADDRESS=(PROTOCOL=TCP)(HOST=10.107.146.71)(PORT=1521))
(CONNECT_DATA=(SERVICE_NAME=TIBQADB)))
```

Failover mode is supported with the basic and preconnect methods.

The sample connection string for an OCI driver is:

```
jdbc:oracle:oci:@net_service_name
```

When using an OCI driver, Transparent Application Failover (TAF) is supported.

You can configure Single Client Access Name (SCAN) for Oracle database in a cluster with numerous nodes. SCAN is a feature of Oracle Real Applications Clusters (RAC) 11g, which provides a single name for clients to access Oracle Databases running in a cluster. You can configure SCAN during the installation of Oracle Grid Infrastructure. Once configured, application tier connection descriptors specify the SCAN name instead of all the virtual hosts in the cluster.

You can specify the SCAN name as follows: `VISION = (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp) (HOST=C-SCAN) (PORT=1521)) (CONNECT_DATA=(SERVICE_NAME=VISION)))`

For more information, see Oracle documentation.

For a thin driver, the sample SCAN connection string is as follows:

```
jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=enrac-scan)(PORT=1521))
(CONNECT_DATA
=(SERVER = DEDICATED)(SERVICE_NAME = TIBQADB)))
```

For an OCI driver, the sample SCAN connection string is as follows:

```
jdbc:oracle:oci:@TIBQADB
```

The sample `TNSNames.ora` file for the above URL is as follows:

```
TIBQADB =
(DESCRIPTION =
(ADDRESS_LIST =
(ADDRESS = (PROTOCOL = TCP)(HOST = enrac-scan)(PORT = 1521))
```

```

)
(CONNECT_DATA =
(SERVICE_NAME = TIBQADB)
)
)

```



For details of the Database URL of a shared JDBC Connection, see *TIBCO BusinessEvents Developer's Guide*.

Other Information about Oracle Database and Drivers

A few helpful points about Oracle database and drivers.

- Use `ojdbc7.jar` drivers file. You can download this file from [Oracle.com](https://www.oracle.com).
- Maximum length for an Oracle table name or column name is 30 characters.
- When OCI client is configured with 32-bit client, Oracle might display "No data found" error. To resolve the error, reduce the recovery batch size using the `be.engine.cluster.recovery.batchsize` property.

For example, if you are getting "No data found" error when the batch size is 10000, a batch size of 5000 might resolve the error, that is,

```
<property name="be.engine.cluster.recovery.batchsize" value="5000"/>
```

Configuring CDD for Special Cases (As Needed)

This section summarizes CDD configuration that may be required to handle special cases.

See [Ontology Identifiers that Exceed the DBMS Maximum Column Length](#).

The CDD file is only required during backing store setup if configured for these special cases. Other aspects of CDD configuration are ignored. Do this aspect of CDD configuration before you use the `be-jdbcdeploy` utility.

Procedure

1. In TIBCO BusinessEvents Studio, open the project's CDD file and select Backing Store from the list on the left.
2. In the tree on the left, click **Overrides** and click **Add**.
3. Select the entity or entities you want to customize and click **OK**.
4. In the tree on the left, select the `/uri` entry for each selected entity in turn, and configure the settings on the right:
 - To exclude an entity from the backing store, uncheck the Has Backing Store checkbox.



In Memory Only mode, if you configure any entity override Mode setting as Memory Only, then backing store is disabled for that entity.

- To specify a short table name for entities whose table names would exceed the database product's maximum length, check the Has Backing Store check box, and enter the name in the Table Name field. See [Ontology Identifiers that Exceed the DBMS Maximum Column Length](#) for details, and for an alternative way to handle long names.



It is recommended that you specify table names that start with "D_" to match the standard naming convention.

- To specify the length of string properties that exceed 255 characters (that is the actual contents stored in the column is more than 255 characters), check the Has Backing Store check box, and in the Properties Metadata section, Max Length setting, specify the expected maximum length for all such properties. See [String Properties that Exceed the DBMS Maximum Column Length](#) for details.

5. Save the CDD.

Adding a JDBC Connection Resource (Now or Later)

Add a JDBC Connection resource to your project and configure it to connect to the backing store. You can do this before or after you configure the database itself. These settings are ignored by the setup utility during the database setup process.

Details below use Oracle 10g database as an example. Adapt the instructions as needed for your database. .



- The value of the **CDD Cluster tab > Backing Store > Connection > Max Size field** overrides the value of the JDBC Connection Resource Maximum Connections setting.
- For design-time features, such as the Test Connection feature, to work correctly, you must add your DBMS product's libraries to the TIBCO BusinessEvents Studio classpath.
- For correct runtime operation, see *TIBCO BusinessEvents Administration*.

Procedure

1. In TIBCO BusinessEvents Studio, open your project, and open the folder where you keep shared resources and select **New > Other > TIBCO Shared Resources > JDBC Connection** .
2. At the New JDBC Connection Wizard dialog, provide a name for the shared connection and click **Finish**.
You will see the JDBC Connection resource editor.
3. In the JDBC Driver field, select the driver for your database from the selection box on the right, for example, **oracle.jdbc.OracleDriver**.
The driver name appears in the box, and a Database URL format appears to the left of the selection box, according to the driver you selected. For the Oracle thin driver, the format is:
`jdbc:oracle:thin:@<host>:<port#>:<db_instancename>`
Check the product readme file to ensure you are using the correct database and driver versions.
4. In the Database URL field, configure the provided format. For example:
`jdbc:oracle:thin:@localhost:1521:ORCL`
where 1521 is the default port, and ORCL is the default instance name for Oracle Database 10g.
5. In the User Name and Password fields, enter the username and password of the database user (see [Run the Initialize Database Script as the DBA or System User](#)).
6. Save the resource.



After you have configured the database and it is running, you can test the connection. Click the **Test Connection** button. If the database is running and the details are correct, you see a success message.

Before you can use this feature, you must first add the JDBC driver to the project **Properties > Build Path > Java Libraries** area in TIBCO BusinessEvents Studio. See note in introduction).

Configuring Backing Store Settings in the CDD (Now or Later)

You can do the backing store configuration before or after you configure the database itself. These settings are ignored by the setup utility.

Reference Tables

Reference tables for all the properties are documented here:

[CDD Cluster Tab Backing Store Settings Reference](#)

[CDD Cluster Tab Backing Store Properties Reference](#)

Procedure

1. In the CDD editor, select Cluster tabObject ManagementBacking Store.
2. To enable backing store functionality, check the **Enabled** check box.
3. In the Database Type field, select **oracle**, **mysql**, **sqlserver**, or **postgres**, depending on the type of database you are using.
4. If you select **oracle** then in the Strategy setting, also select the pooling strategy: **oracle** or **jdbc**. Connection pool settings are interpreted differently depending on your choice here, as documented in the reference tables listed above.
5. Choose the database write strategy:
 - To use the cache-aside strategy, check the **Cache Aside** check box.
 - To use the write behind aside strategy, uncheck the **Cache Aside** check box.

See Post-RTC and Epilog Handling and Tuning Options in *TIBCO BusinessEvents Architect's Guide* for more on write strategies.
6. If you want to enforce JDBC connection pool settings and properties, check the **Enforce Pools** check box. If you do not want to enforce connection pool settings and properties, uncheck the box.
7. Select **Connection** on the left do the following:
 - Select the JDBC Connection shared resource configured for the backing store.
 - Set the minimum, maximum, and initial sizes of the JDBC connection pool as needed.
 - Add additional properties to the Cluster properties sheet as needed (for example, if you are using Oracle Strategy). See [CDD Cluster Tab Backing Store Properties Reference](#) for details.
8. Configure domain object overrides

As desired, you can configure CDD settings related to domain objects (entity instances) such as mode, preloading, limited cache options.

See [Domain Objects Configuration](#).



If you change settings that enable or disable the backing store for individual entity types after you have created the backing store schema, you must update the schema, as explained in [Updating Existing Backing Store Schema](#).

Preventing Database Outages when a Cluster Ceases with Processing

A new feature was added to allow the Inference engine to continue to run during a database down period, when all database updates are buffered through the ActiveSpaces caching mechanism. Although the length of the database down period is not specified, it is limited by the available memory for ActiveSpaces datagrid (or by the buffer sizes in case of Coherence).

When database connections are restored, all buffered transactions will play back and data loss will be avoided.

Required Configuration Settings

For this feature to work, the following configuration settings are required. These settings ensure that all the data that the Inference engine needs is already cached, and anything that is not found in the cache will not be in the database either. Regardless whether or not database is available, if a 'read/get' does not find the searched entity in cache, the database is never queried.

1. Shared-All persistence with Write-Behind (Cache Aside=false)
2. Persistence Mode is set to ASYNC
3. Unlimited cache (both entity and object-table, even if object-table is disabled
useobjecttable=false)
4. Preload and Recover all caches (both entity and object-table, even if object-table is disabled
useobjecttable=false)
5. ObjectCacheFullyLoaded flag is set to true:

```
<property name="be.engine.cluster.isObjectCacheFullyLoaded" value="true"/>
```

Suggested Settings

In addition to the required configuration settings, the following CDD settings are recommended:

1. Cache Agent Quorum = 2 (or more)
2. Number of Backup Copies = 1 (replication of 1 or more)
3. Disable connection checking in Inference agents (prevents exceptions in inference logs

```
<property name="be.backingstore.connection.retry.count" value="0"/>
```


and with cache-aside database connections are not needed except during startup).



If Inference engine is configured to be used as scheduler, then skip this setting. Cleanup object-table entries marked for deletion during startup (manually or automatically by setting

```
<property name="be.engine.cluster.cleanup" value="true"/>
```

4. Cleanup object-table entries marked for deletion during startup (manually or automatically by setting

```
<property name="be.engine.cluster.cleanup" value="true"/>
```
5. During testing set log-configuration roles as the following (this will help with debugging)

```
<roles>*:info runtime.service:info kernel.core:debug backingstore:all  
jdbcstore:all jdbcstore.impl:all sql.text:all sql.vars:all</roles>
```

Limitations

Scheduler (DB Poller) is database dependent and will not continue executing while database is down.



If cache server quorum is breached before full recovery, data loss is unavoidable. If every cache servers needs to be shut down, all the changes done during the database down period will be lost.

Testing Recommendations

- During testing, test disconnects by 'shutting down database service and machine', 'disconnecting database server from network' and by other means possible.
- Test for both very short outages (30 seconds), very long outages (hours) in-between outages, and repeated outages.
- Start tests first with a project where entities are created only, then later test when entities are deleted only and finally test the case where entities are created, modified and deleted as usual.

- If an entity is first created with ActiveSpaces and then deleted during a database outage period, they cancel each other out. As a result, there will be no related database transaction when the connection is restored.
- If an entity is modified multiple times with ActiveSpaces during a database outage period, only the last update will be kept. As a result, there will be only be a single database transaction when the connection is restored.
- When a database is disconnect with ActiveSpaces, there will be exceptions in the cache engine logs. These exceptions should almost immediately suspend the 'Persister' involved. The space will enter the 'Persister State = offline' state. This can also be accomplished by issuing the as-admin command "suspend persistence 'dist-unlimited-bs-Test--be_gen_Concepts_***'".
- When the database connections are re-established with ActiveSpaces, it will resume with all 'Persisters'. The space will enter the 'Persister State=replaying/online' state. This can also be accomplished by issuing the as-admin command "resume persistence 'dist-unlimited-bs-Test--be_gen_Concepts_***'".
- The ActiveSpaces behavior can also be tested in 'isolation' without disconnecting the database. You can instead suspend and resume commands for all the relevant spaces in the cluster.
- During the database outage, you may notice that the engine throughput will increase beyond normal. This is because all the database transactions are deferred (essentially, the system runs only in Cache mode with no persistence). When the connection is re-established throughput will decrease during replay period.
- With ActiveSpaces, monitor the space "ToPersist" count by using the as-admin tool. This shows the in-flight updates in cache which are not yet persisted.

Building the EAR File

The EAR file is required when you run the database setup utility. Ontology information in the EAR is used to build tables in the database.

Ensure that the project ontology is configured correctly and is complete. Ontology information is required by the database setup utility. Other aspects of TIBCO BusinessEvents Studio configuration are ignored by the database setup utility and can be done before or after you run the utility.

Procedure

1. In BusinessEvents 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, you can change the EAR file name, specify the location, and complete any other settings as desired.
2. Click **Apply**, then click **OK**. You see messages as the EAR file builds, then you see a message that the EAR file has built correctly:

Initializing the Database and Generate Non-Project Tables

Ensure that you have done all earlier tasks that may pertain to your case.

See [Ontology Identifiers that Exceed the DBMS Maximum Column Length](#).

Run the Initialize Database Script as the DBA or System User. This script creates the TIBCO BusinessEvents user and initializes the database.



Running the script `initialize_database_YourDBMS.sql` script deletes the user before creating it again. Running the script `create_tables_YourDBMS.sql` drops all database tables before creating them again. This means you can run these scripts again during test phases of your project development, without having to take extra cleanup steps.

The first time you run the `create_tables_YourDBMS.sql` script, you see harmless error or warning messages because there is nothing to delete.

If you are updating the schema for an existing backing store, see [Updating Existing Backing Store Schema](#).

Procedure

1. As desired, change the default TIBCO BusinessEvents user credentials: Open the `initialize_database_YourDBMS.sql` script for editing and change the default username and password. The documentation uses the default username (BE_USER) and password (BE_USER)
2. Open a command window in the `BE_HOME/bin` directory (default location of the scripts), and the appropriate command for your DBMS at the prompt:

- For Oracle:

```
sqlplus sys_user/sys_user_password@SID @initialize_database_oracle.sql
```

Type `exit` to exit and commit.

- For SQL Server:

```
osql -S Your-Server -U sys_user -P sys_user_password -n -i  
initialize_database_sqlserver.sql
```

- For MySQL:

```
mysql -u username -p database_name < initialize_database_mysql.sql
```

- For PostgreSQL:

```
use database_name  
psql -u username -f initialize_database_postgres.sql
```

This script creates the TIBCO BusinessEvents database user. This user must be used to run the other scripts. You see messages like the following:

```
User dropped.  
User created.  
Grant succeeded.
```



Using your database product, you can configure additional users to access the database, in addition to this user.

Run the Create Tables Scripts as the TIBCO BusinessEvents User

Log on as the TIBCO BusinessEvents user, `BE_USER` by default and run a script to create non-project specific tables.

3. Open a command window in the `BE_HOME/bin` directory (default location of the scripts), and type the appropriate command for your DBMS at the prompt:

- For Oracle:

```
sqlplus BE_USER/BE_USER@SID @create_tables_oracle.sql
```

Type `exit` to exit and commit.

- For SQL Server:

```
osql -S Your-Server -d Your-DB -U BE_USER -P BE_USER -n -i  
@create_tables_sqlserver.sql
```

- For MySQL:

```
mysql -u BE_USER -p database_name < create_tables_mysql.sql
```

- For PostgreSQL:

```
use database_name
psql -u username -f create_tables_postgres.sql
```

Use the credentials are defined in the `initialize_database_oracle.sql`, `initialize_database_sqlserver.sql`, and `initialize_database_mysql.sql` files. By default those are: username `BE_USER`, with password `BE_USER`.

Result

You see various harmless error messages the first time you run the script.



TIBCO BusinessEvents Studio provides a wizard to configure the required properties and generate the project-schema-specific SQL scripts.

Project-Schema-Specific SQL Scripts

You can do this task using a TIBCO BusinessEvents Studio wizard, or you can do it manually.

Before you Begin



- Ensure that you copied your JDBC drivers file to `BE_HOME/lib/ext/tpcl` (or other location in your class path).
- Ensure the DBMS is started.

Generating Scripts Using the JDBC Deployment Wizard

You can generate the project-schema-specific SQL scripts using the JDBC Deployment wizard in TIBCO BusinessEvents Studio.

Generating the Project-Schema-Specific SQL Scripts (with Wizard)

Procedure

1. In TIBCO BusinessEvents Studio Explorer, right-click the name of the project for which are creating SQL scripts and select **Export > TIBCO BusinessEvents > JDBC Deployment**.
2. Click **Next**.
You see the **Generate JDBC deployment scripts** wizard.
3. Complete the values as follows:

Field Name	Description
Database Type	Select the type of database you are using from the drop-down list. Choose either <code>oracle</code> , <code>mysql</code> , <code>sqlserver</code> , or <code>postgres</code> . The default value is <code>oracle</code>
Generate ANSI Scripts	Select this check box to use ANSI compatible SQL types during script generation. Default setting is checked.
Generate Optimized Scripts	Select this check box to eliminate schema generation for in-memory events.

Field Name	Description
Use Extended Max String Size	This is active only for the <code>oracle</code> database type. Sets the maximum string size value to extended, that is to 32767 bytes.
Cluster Deployment Descriptor	Browse to and select the CDD file you want to use for the project.
Output Directory	Browse to and select a directory where the scripts are to be generated, for example, <code>BE_HOME/bin</code> (This directory is used if you generate files manually.)
Output Script Name Prefix	Enter a prefix for the output script filenames. For example, if you enter <code>acme</code> , the following scripts are generated: <code>acme.sql</code> , <code>acme.aliases</code> , <code>acme_alter.sql</code> , <code>acme_cleanup.sql</code> , <code>acme_delete</code> , and <code>acme_remove.sql</code> .

4. Do one of the following:
 - If you are creating a new JDBC backing store, click **Finish**. The scripts are generated in the location you specified. Information about script generation is also printed to the Console tab.
 - If you are migrating an existing JDBC backing store, click **Next**. You see the Generate Migration SQL Scripts page.
5. On the Generate Migration SQL Scripts page, complete the values as follows.

Field Name	Description
Generate Migration Scripts	Check this check box to generate SQL scripts to migrate an existing backing store. Once checked, the rest of the fields on this page are enabled. The default value is unchecked.
Connection Configuration	Browse to and select the JDBC Connection shared resource used to connect to the existing backing store.
Database URL	Enter the database URL that points to the existing backing store.
Database Username	Enter the database username that was used when setting up the existing backing store.
Database Password	Enter the database password that was used when setting up the existing backing store.
Database Schema Owner	Enter the database schema owner for the existing backing store.
Test Connection	Click Test Connection to test if a success

Generating Scripts Manually

This section explains how to generate scripts using a manual procedure, instead of using the TIBCO BusinessEvents Studio wizard.



If you are using the SQL Server, then before you execute **be-jdbcdeploy**, open the file `BE_HOME/bin/be-jdbcdeploy.tra` for editing. Specify `sqlserver` in the last line:

```
java.property.jdbcdeploy.database.type [oracle | sqlserver | mysql | postgres]
```

This step requires the EAR file for the project. The database utility uses the project ontology information from the EAR file.

Procedure

1. Open a command window and navigate to `BE_HOME/bin`.

2. Run `be-jdbcdeploy.exe` using a command with the following format:

```
be-jdbcdeploy [-h] [--propFile] [-p property file] [-o schema output file] [-c CDDpath] [-a true|false] [-optimize] [-maxstringsize standard|extended] EAR Path
```

For example:

```
be-jdbcdeploy -o acme -c D:/myproj/acme.cdd D:/ears/acme.ear
```

3. The generated scripts appear in the directory where you run the executable.
For example, if you provide the schema output filename `acme`, you would see files called `acme.sql`, `acme.aliases`, `acme_alter.sql`, `acme_cleanup.sql`, `acme_delete`, and `acme_remove.sql`.

The user-defined part of the database schema is in the schema output file (*yourname.sql*) as schema definition commands. The options are explained in [Schema Definition Commands Options](#)

Schema Definition Commands Options

Run the script to build the schema in the database with the provided options.

Option	Description
<code>--propFile</code> , <code>/--propFile</code> , or <code>-system:propFile</code>	Optional. When you execute <code>be-jdbcdeploy</code> , it searches for a property file of the same name in the working directory. This property file provides startup values and other parameters to the executable. You can specify the path and filename of a startup property file explicitly using the <code>--propFile</code> parameter. For example, if you start the engine from a directory other than <code>BE_HOME/bin</code> , then you would generally use <code>--propFile</code> to specify <code>BE_HOME/bin/be-jdbcdeploy.tra</code> .
<code>-p</code> , <code>/p</code> , <code>-property</code> , or <code>/property</code>	Optional. Allows you to pass one or more supplementary property files to <code>be-jdbcdeploy</code> . Specify the path and filename. Values in supplementary property files override the values in the startup property file.
<code>-o</code> , <code>/o</code> , <code>-out</code> , or <code>/out</code>	Required. Specifies the schema output filename for deployment. Tip If you specify a directory path, the backing store scripts are generated in the specified directory and the last element of the path is taken as the schema output filename.

Option	Description
-c, /c, -cdd, or /cdd	Optional. Specifies the absolute path to the CDD file. See Ontology Identifiers that Exceed the DBMS Maximum Column Length to understand when the CDD file is needed.
-optimize	Optional. Use this option to eliminate schema generation for in-memory events.
<i>EARpath</i>	Required. The last option is always the EAR file path.
-h, /h, -help, or /help	Displays this help.
-a, /a, -ansi, or /ansi	Optional. If set to true, ANSI compatible SQL types are used during script generation. Allowable values are true and false. For ANSI compatible databases, it is set to true by default.

Aliases File and Project Schema Script

For every entity, property, or state machine whose database identifier name exceeds the database maximum length, a table name entry is created in the generated *yourname.aliases* file (for example, *acme.aliases*).

See [Ontology Identifiers that Exceed the DBMS Maximum Column Length](#) for more information, and for an alternative way to specify short table names.

It's a good idea to check the aliases file for entries, even if the TIBCO BusinessEvents names are not very long. The length of the generated database table names is not easy to predict.

Optionally, you can edit the file to provide more meaningful names.



It is recommended that you keep the aliases file for future reference. If the project ontology changes after the backing store has data in it, you must also update the database schema to match the new schema (as explained in [Updating Existing Backing Store Schema](#)). If you modified the generated aliases, you must use the same aliases again when you update the schema, to preserve those columns and their data.

Key word mapping file

Entries in the key word mapping file are also added to the aliases file so you can replace the key word aliases with project-specific ones, as desired (generally in a second pass). For details see [Step 3 f Needed — Map Key \(Reserved\) Words to Aliases](#).

Configuring Aliases File and Project Schema Script

To configure aliases files and project schema scripts:

Step 1 Check the Aliases File and Modify Aliases as Desired

Procedure

1. Open the *yourname.aliases* file for editing.
2. Replace any aliases as desired with more meaningful short names.
Make sure that each name is unique. It's a good idea to leave any system generated prefixes or suffixes in place for consistency of names across the database.

3. Re-run the `be-jdbcdeploy` tool, using the same parameters as before.
For details see [Generate the Project-Schema-Specific SQL Scripts \(with Wizard\)](#). This time, the aliases you created are used.

Step 2 Run the Project Schema Script (as BE_USER)

In this step, you log on as the user you created and run a script to create the project related part of the database schema.

Procedure

- Open a command window in `BE_HOME/bin` and run the `yourname.sql` script. (for example, `@acme.sql`). Use the user `BE_USER`, password `BE_USER` (or whatever username and password you have set up).
 - For Oracle:

```
sqlplus BE_USER/BE_USER @ yourname.sql
```
 - For SQL Server:

```
osql -S Your-Server -d Your-DB -U BE_USER -P BE_USER -n -i @yourname.sql
```
 - For MySQL:

```
mysql -u BE_USER -p database_name < yourname.sql
```
 - For PostgreSQL:

```
use database_name  
psql -u username -f yourname.sql
```

Result

If there are no errors, your database tables are now configured for use. If there are errors you may need to add some mappings to the key word mapping file.

Step 3 If Needed - Map Key (Reserved) Words to Aliases

This task is optional and performed in case there are errors in the previous steps.

Complete this task only if you saw errors after completing [Step 2 Run the Project Schema Script \(as BE_USER\)](#). Such errors are caused when your project ontology uses terms that are key words (reserved terms) in the DBMS you are using. You must map these terms to an alias in the keyword mapping file.

Procedure

1. Edit the `BE_HOME/dbkeywordmap.xml` file to add entries. Below is the format followed by an example:

```
<keyword name="dbKeyword" mapname="nonDbKeyword"/>  
  <keyword name="start" mapname="start_"/>
```
2. Repeat [Generate the Project-Schema-Specific SQL Scripts \(with Wizard\)](#), and tasks following as needed.
3. If desired, create project-specific aliases for the key word mappings as explained in [Providing Project-Specific Keyword Aliases](#).

Providing Project-Specific Keyword Aliases

When you repeat , the new key words are added to the *yourname*.aliases file. You can create project-specific aliases for the key word mappings as desired. Then repeat again and continue.



You must generate the SQL scripts a minimum of three times if you add keyword mappings to the aliases file — it might be more because you may not catch all errors at once. For example, if there are multiple keyword clashes in one table, only the first are reported. Perform this loop until no more errors occur.

Procedure

1. Generate the SQL scripts and run them (as explained in the procedures). Errors occur due to key word clashes.
2. Add the appropriate key word mapping entries to the key word mapping file.
3. Generate the SQL scripts again.
4. To use project-specific aliases for the keyword mappings (Optional):
 - a) Edit the aliases file entries for the key word mappings.
 - b) Generate the SQL scripts again.
5. Run the SQL scripts to create the backing store.

Step 4 Project Configuration (As Needed)

Complete the TIBCO BusinessEvents Studio project configuration tasks if you have not already done so. These tasks can be done before or after database setup. See the following sections:

- [Adding a JDBC Connection Resource \(Now or Later\)](#)
- [Configuring Backing Store Settings in the CDD \(Now or Later\)](#)



Update your schema if your ontology changes, or if you want to include or exclude different entities in the backing store. See [Updating Existing Backing Store Schema](#) for details.

Updating Existing Backing Store Schema

If you change the project ontology, that is, if you create, alter or delete a concept or an event, you must update the backing store schema so it matches the updated ontology. In the case of changes in project ontology, you must update the backing store schema before you deploy the updated project.

You may also wish to change which entities are excluded from the backing store using CDD settings (see [Excluding Entities from the Backing Store](#) for more information. This change does not require project redeployment. It requires that the updated CDD file is copied to all runtime machines.

Examine the alter script before you run it. The section [What the Schema Update Utility Can Handle Automatically](#) provides more information.

To run the Schema Update Utility:

Procedure

1. Gracefully shut down the deployed application (all agents including cache agents).
2. Back up your existing database.
3. Generate the updated EAR file for the modified project.
4. If you modified aliases when you created the schema, locate the *yourname*.aliases file you used. It will help you to modify those aliases in the newly generated file, so they match.

5. Open the `be-jdbcdeploy.tra` file for editing and set the following properties:
 - a) `be.jdbc.schemamigration.url=DbURL`
 - b) `be.jdbc.schemamigration.user=username`
 - c) `be.jdbc.schemamigration.pswd=password`
6. Use one of the following:
 - a) Database URL that points to the existing backing store. See [Adding a JDBC Connection Resource \(Now or Later\)](#) for example URLs.
 - b) Same username and password you used when setting up the backing store. See [Run the Initialize Database Script as the DBA or System User](#)

These properties enable the program to compare the schema of the existing database with the ontology in the project EAR file, and generate the alter script.

7. Log on as the user name you specified in [Run the Initialize Database Script as the DBA or System User](#).
8. Run the `be-jdbcdeploy` utility as explained in [Generate the Project-Schema-Specific SQL Scripts \(with Wizard\)](#), using the *updated* EAR file.
9. If any of the new or changed definitions result in entries in the `yourname.aliases` file, and you want to change the provided aliases, follow instructions in [Configuring Aliases File and Project Schema Script](#). If you modify aliases, remember to generate the scripts again so the modified aliases are used.



You must use the same aliases that you used before. If any were modified when the schema was created, you must modify them the same way when updating the schema. It can be useful to refer to the original aliases file.

10. Examine the generated `yourname_alter.sql` script and modify as needed so you only run statements for changes you want to make. See [What the Schema Update Utility Can Handle Automatically](#) for details.
11. Run the `yourname_alter.sql` script.

Result

Your database tables are now configured for use.

What the Schema Update Utility Can Handle Automatically

You must examine the alter script before you run it. Decide what changes to make manually and what changes to make using the script, taking into account the kind of data in the tables. Entries that could result in data loss are commented. Remove or comment entries for changes you will make manually.

Adds

The schema migration utility handles addition of entity types and attributes. New entity types and attributes are added to the database schema.

Changes (Drop and Add) — Assess individually

The utility handles changes to attributes (entity properties) as DROP and ADD operations. However, DROP operations are commented in the script to avoid data loss.

If a column is empty, or you do not want to keep the data they contain, you can enable the DROP operation and let the utility handle the change.

If the column contains data that you want to keep, then make the change manually using an appropriate database tool. For example, you can change the data type of a column from string to double without loss of data, as long as all the column values are numeric values.

Entity Deletions

If an entity is deleted from the TIBCO BusinessEvents Studio project, the corresponding tables are not dropped from the database schema. Existing data is not lost. Deleted entities are not mentioned in the alter script. Manually keep track of and delete such tables as needed.

Attribute Deletions

The schema update utility does handle deletion of entity attributes. SQL statements for deleted attributes are generated but they are commented. Examine the alter script and enable these commands if you want to execute them. Note that existing data is lost when you drop an attribute

Example Alter Script

Below is an example *yourname_alter.sql* script.

Property type change

```
-- ##### WARNING : Non-alterable Ontology changes found. Please see following
-- errors. Manual schema-migration is required.
--* For Concept Concept1 field PROPERTY_1 type changed from VARCHAR2 to LONG
-- ALTER TABLE D_Concept1 DROP ( Property_1 );
ALTER TABLE D_Concept1 ADD ( Property_1 numeric(19) );
New table
DROP TABLE D_Book_rrf;
CREATE TABLE D_Book_rrf (pid numeric(19), propName char varying(255), id$
numeric(19) not null);
New property
-- ALTER TABLE D_MyConcept DROP ( FOLDER_1 );
ALTER TABLE D_MyConcept ADD ( Folder_0 char varying(255) );
```

Backing Store Table Reference

The backing store uses relational tables and SQL data types for ease of maintenance. The SQL (DDL) scripts use ANSI SQL type definitions (where supported by the target DBMS product).

Each ontology type in the backing store has its own primary table and zero or more second-level tables. There are only two levels of tables, which makes the database easier to manage and easier to understand. Because the backing store adheres to SQL standards and a straight-forward structure, standard database tools can be used to view backing store data.

Primary Tables

Primary tables contain only primitive properties such as the following:

Property	Note
cacheId	Entity version number (starts with 1)
time_created\$	Time when the entity was created
time_last_modified\$	Time when the entity was last modified
parent\$_id\$	Id of the parent for contained concepts
id\$	Unique Id of the entity (must be unique across all entities)
extId\$	Unique (or null) extId assigned
state\$	Always set to 'C' meaning 'Created' (reserved for future use)

Secondary Tables

Secondary tables are used for complex properties, that is, arrays, properties with history, and concept relationship properties. Each array and history-enabled property has a separate table. Only primitive properties are stored in the primary table.

Secondary table structure

property Type	Column	Description
Array	pid\$	Parent ID
	valPid\$	Array index
	val	Item's value
History	pid\$	Parent ID
	howMany	Number of history items
	timeIdx	Item's time stamp
	val	Item's value
Array with History	pid&	Parent ID
	valPid\$	Array index
	howMany	Number of history items
	timeIdx	Item's time stamp
	val	Item's value

Reverse Reference Tables

Each concept also has a reverse reference table. This table's name contains the concept name and ends with the characters `_rrf$`. It has these columns:

Column	Description
pid\$	Parent ID from the main concept table
propertyName\$	Property name (field) from the referencing concept.
id\$	Identifier (id\$) of the referencing concept.

Class-to-Table Mapping

This table contains the mapping between class names and table names, and the mapping between complex property field names and secondary table names. For example:

```
'be.gen.Ontology.DeleteVerifyEvent', 'D_DeleteVerifyEvent'
'be.gen.Ontology.Treatment', 'D_Treatment'
```

```
'be.gen.Ontology.Treatment', 'rrf$', 'D_Treatment_rrf$'  
'be.gen.Ontology.BaseAlert', 'treatments', 'D_BaseAlert_treatments'
```