TIBCO ActiveMatrix® Runtime UDDI Server

User's Guide

Software Release 3.1
March 2010

Important Information

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

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

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

TIB, TIBCO, TIBCO Adapter, Predictive Business, Information Bus, The Power of Now, TIBCO ActiveMatrix are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

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

THIS SOFTWARE MAY BE AVAILABLE ON MULTIPLE OPERATING SYSTEMS. HOWEVER, NOT ALL OPERATING SYSTEM PLATFORMS FOR A SPECIFIC SOFTWARE VERSION ARE RELEASED AT THE SAME TIME. SEE THE README.TXT FILE FOR THE AVAILABILITY OF THIS SOFTWARE VERSION ON A SPECIFIC OPERATING SYSTEM PLATFORM.

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

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

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

Copyright © 2008-2010 TIBCO Software Inc. ALL RIGHTS RESERVED.

TIBCO Software Inc. Confidential Information

Contents

Figures

Preface

TIBCO ActiveMatrix Runtime UDDI Server provides an integrated UDDI registry that can be used by TIBCO ActiveMatrix products or other products that use the standard UDDI v3 API.

This manual describes how to use the user interface, the command line, and the API of TIBCO ActiveMatrix Runtime UDDI Server.

Topics

Related Documentation

This section lists documentation resources you may find useful.

TIBCO ActiveMatrix Runtime UDDI Server Documentation

The following documents form the ActiveMatrix Runtime UDDI Server documentation set:

- *TIBCO ActiveMatrix Runtime UDDI Server Installation* Read this manual for instructions on site preparation and installation.
- *TIBCO ActiveMatrix Runtime UDDI Server Administrator's Guide* Read this manual for instructions on configuring the product.
- *TIBCO ActiveMatrix Runtime UDDI Server User's Guide* Read this manual for instructions on using the product.
- *TIBCO ActiveMatrix Runtime UDDI Server Release Notes* Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

Third-Party Documentation

The following documents are related to the TIBCO ActiveMatrix Runtime UDDI Server.

- *UDDI Spec V3.0* This document describes the web services and behaviors of all instances of a UDDI registry. It is available from the following web site: http://uddi.org/pubs
- *Oracle Database Administrator's Guide* This manual provides information about operating an Oracle database. It is available from the Oracle web site: http://download.oracle.com/docs/cd/B14117_01/server.101/b10739.pdf
- *MySQL 5.1 Reference Manual* This manual provides information about operating a MySQL database. It is available from the MySQL web site: http://dev.mysql.com/doc/refman/5.1/en/
- *Server Products and Technologies for SQL Server* This online book provides information about operating a SQL Server database. It is available from the SQL Server web site: http://technet.microsoft.com/en-us/library/default.aspx
- *Sybase Adaptive Server Enterprise (Archive)* This online book provides information about operating a Sybase ASE database. It is available from the Sybase web site:

http://sybooks.sybase.com/nav/summary.do?prod=9938&lang=en&prodName=Adaptive+Server+Enterprise&archive=0&Submit.x=25&Submit.y=16

- *IBM DB2 Information Center* You can find information about operating an IBM DB2 database from the IBM web site: http://publib.boulder.ibm.com/infocenter/db2luw/v8/index.jsp
- *HSQLDB User Guide* This manual provides information about operating an HSQLDB. It is available from the HSQLDB web site: http://www.hsqldb.org/doc/guide/
- *JBoss Application Server Documentation* You can find information about configuring the JNDI resource for the JBoss application server in the documentation. It is available from the JBoss web site: http://www.jboss.org/jbossas/docs/.
- *Apache Tomcat 6.0 JNDI Resources HOW-TO* This online book provides information about configuring the JNDI resource for the Apache Tomcat server. It is available from the Apache Tomcat web site: http://tomcat.apache.org/tomcat-6.0-doc/jndi-resources-howto.html
- *IBM WebSphere Application Server, Version 7.0 Information Center* You can find information about configuring the JNDI resource for the IBM Websphere application server from the IBM web site: http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp

Typographical Conventions

The following typographical conventions are used in this manual.

Table 1 General Typographical Conventions

Convention	Use
TIBCO_HOME *ENV_HOME* *UDDI_HOME*	Many TIBCO products must be installed within the same home directory. This directory is referenced in documentation as *TIBCO_HOME*. The value of *TIBCO_HOME* depends on the operating system. For example, on Windows systems, the default value is `C:\tibco`. Other TIBCO products are installed into an installation environment. Incompatible products and multiple instances of the same product are installed into different installation environments. The directory into which such products are installed is referenced in documentation as *ENV_HOME*. The value of *ENV_HOME* depends on the operating system. For example, on Windows systems the default value is C:\tibco. TIBCO ActiveMatrix Runtime UDDI Server installs into a directory within *TIBCO_HOME*. This directory is referenced in documentation as *UDDI_HOME*. The value of *UDDI_HOME* depends on the operating system. For example on Windows systems, the default value is `C:\tibco\RuntimeUDDIServer\3.1`
`code font`	Code font identifies commands, code examples, filenames, pathnames, and output displayed in a command window. For example: Use `MyCommand` to start the foo process.
`bold code font`	Bold code font is used in the following ways: • In procedures, to indicate what a user types. For example: Type **`admin`**. • In large code samples, to indicate the parts of the sample that are of particular interest. • In command syntax, to indicate the default parameter for a command. For example, if no parameter is specified, `MyCommand` is enabled: `MyCommand [`**`enable`**` | disable]`

Table 1 General Typographical Conventions (Cont'd)

Convention	Use
italic font	Italic font is used in the following ways: • To indicate a document title. For example: See *TIBCO ActiveMatrix BusinessWorks Concepts*. • To introduce new terms For example: A portal page may contain several portlets. *Portlets* are mini-applications that run in a portal. • To indicate a variable in a command or code syntax that you must replace. For example: `MyCommand` *PathName*
Key combinations	Key name separated by a plus sign indicate keys pressed simultaneously. For example: Ctrl+C. Key names separated by a comma and space indicate keys pressed one after the other. For example: Esc, Ctrl+Q.
	The note icon indicates information that is of special interest or importance, for example, an additional action required only in certain circumstances.
	The tip icon indicates an idea that could be useful, for example, a way to apply the information provided in the current section to achieve a specific result.
	The warning icon indicates the potential for a damaging situation, for example, data loss or corruption if certain steps are taken or not taken.

Terminology and Acronyms

The following acronyms are used in this manual:

Table 2 Terminology and Acronyms

Acronym	Meaning
API	Application Programming Interface.
GUI	Graphical User Interface.
UDDI	Universal Description, Discovery and Integration.
XML	Extensible Markup Language.

How to Contact TIBCO Support

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

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

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

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

  https://support.tibco.com

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

Tables

Chapter 1 Overview

This chapter introduces TIBCO ActiveMatrix Runtime UDDI Server. It demonstrates ways to interact with TIBCO ActiveMatrix Runtime UDDI Server, including the three built-in features: the HTTP GET service, the category system, and the Polling Subscription API. It also introduces TIBCO ActiveMatrix UDDI Service Console and Ant-based tasks.

Topics

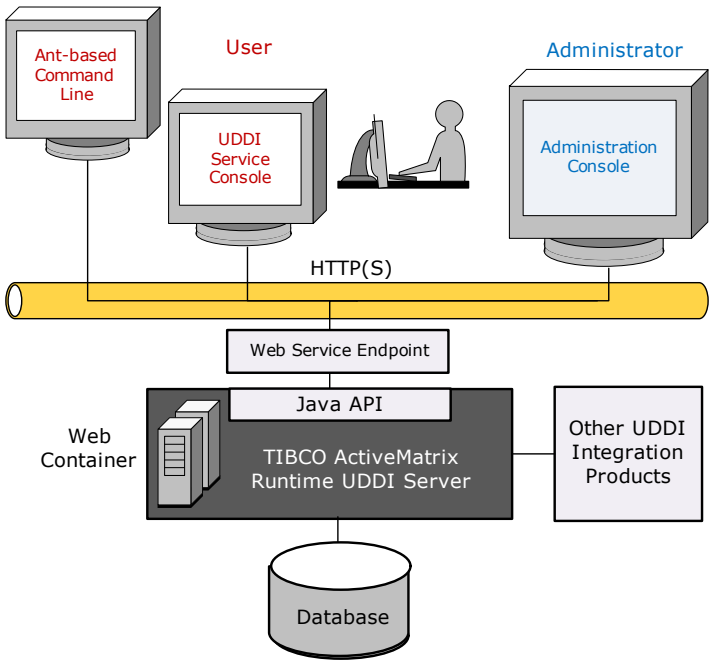
An Introduction to TIBCO ActiveMatrix Runtime UDDI Server

UDDI (Universal Description, Discovery, and Integration) is a standard that enables organizations to publish and discover services. It encourages service reuse.

TIBCO ActiveMatrix Runtime UDDI Server provides an integrated UDDI registry that can be used by TIBCO ActiveMatrix products. There are four major components included in this product: a client with the Ant-based Command Line interface, the web-based UDDI Service Console, the web-based Administration Console, and TIBCO ActiveMatrix Runtime UDDI Server, as shown in Figure 1.

- This release of TIBCO ActiveMatrix Runtime UDDI Server supports a subset of UDDI version 2 and can convert some UDDI version 2 SOAP messages to the UDDI version 3 format. This enables TIBCO ActiveMatrix Runtime UDDI Server to work with TIBCO products that can process web services conforming to UDDI version 2.
- For information about the web-based Administration Console, see *TIBCO ActiveMatrix Runtime UDDI Server Administrator's Guide*.

Figure 1 TIBCO ActiveMatrix Runtime UDDI Server Structure



Interacting with TIBCO ActiveMatrix Runtime UDDI Server

Currently, you can interact with TIBCO ActiveMatrix Runtime UDDI Server through:

- TIBCO ActiveMatrix UDDI Service Console. TIBCO ActiveMatrix UDDI Service Console is a web application that comes with TIBCO ActiveMatrix Runtime UDDI Server. It allows you to perform various TIBCO ActiveMatrix Runtime UDDI Server operations in a user-friendly interface. For more information, see Chapter 2, TIBCO ActiveMatrix UDDI Service Console.
- Ant-based tasks. TIBCO ActiveMatrix Runtime UDDI Server supports multiple types of Ant-based tasks, including find tasks, get tasks, save tasks, delete tasks, subscription tasks, and security tasks. For more information, see Chapter 3, Command Line Interface.
- Polling Subscription API. Polling Subscription API is a TIBCO-proprietary API based on the standard API that manages subscriptions and the start/end time. It provides an easy way to call the `get_subscriptionResults` API. For more information, see Chapter 4, Polling Subscription API.
- HTTP GET service. The HTTP GET service provided by a UDDI node enables you to retrieve XML and HTML representations of UDDI data structures. For more information, see Chapter 5, Using the HTTP GET Service.
- Other TIBCO products with the UDDI Registry function built in, such as TIBCO Business Studio, TIBCO ActiveMatrix Administrator, TIBCO ActiveMatrix BusinessWorks, and TIBCO ActiveMatrix Policy Manager. For more information, see Chapter 6, Integration with Other TIBCO Products.
- Web services provided by TIBCO ActiveMatrix Runtime UDDI Server. Four canonical web services are provided under a special business named `node`. They are UDDI Inquiry Service, UDDI Publication Service, UDDI Security Service, and UDDI Subscription Service. These services can not be modified or deleted.

Default Server

When installing TIBCO ActiveMatrix Runtime UDDI Server, you can choose to install a default server. The default server is on Apache Tomcat 6.0 with the HTTP port *58080* and uses an HSQL database.

For detailed information about the default server, refer to *TIBCO ActiveMatrix Runtime UDDI Server Administrator's Guide*.

Chapter 2 TIBCO ActiveMatrix UDDI Service Console

This chapter describes how to perform different TIBCO ActiveMatrix Runtime UDDI Server operations in TIBCO ActiveMatrix UDDI Service Console. It also contains an example of registering a service with TIBCO ActiveMatrix Runtime UDDI Server.

Topics

Overview

TIBCO ActiveMatrix UDDI Service Console (UDDI Service Console) is a web application that comes with TIBCO ActiveMatrix Runtime UDDI Server. It enables you to perform various TIBCO ActiveMatrix Runtime UDDI Server operations in a user-friendly interface.

This application has been optimized for Microsoft Internet Explorer 8 and Mozilla Firefox 3. Results in other browsers may be unpredictable.

UDDI Objects

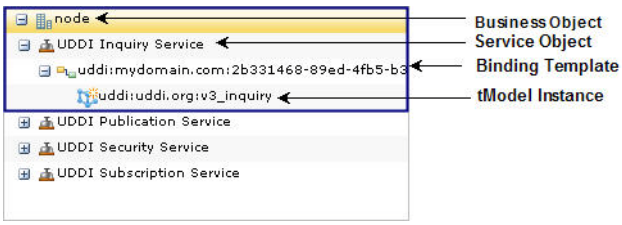
All operations that can be performed in the UDDI Service Console are based on UDDI objects.

The following two types of UDDI objects are used in the UDDI Service Console:

- **Correlated objects**

  Correlated objects include *businesses*, *services*, *binding templates*, and *tModel instances*. They are hierarchically organized. This means that, a business can contain multiple services, a service can contain multiple binding templates, and a binding template can contain multiple tModel instances. After Logging in to Service Console, if you expand a UDDI object tree, you can see these objects structured in it, as shown in Figure 2.

Figure 2 The Correlated Objects



- **Individual UDDI objects**

  In contrast to correlated objects, individual UDDI objects are not hierarchically organized. They are independent of each other. *Subscription*, *category*, and *tModel* are individual objects.

Quick Start

This section uses a sample to describe the basic steps that are required to publish a service in TIBCO ActiveMatrix Runtime UDDI Server through UDDI Service Console.

1. Logging in to Service Console, page 7
2. Creating a tModel Containing the WSDL Document, page 7
3. Creating a Business, page 9
4. Creating a Service, page 9
5. Creating a Binding Template, page 9
6. Creating a tModel Instance, page 10

Logging in to Service Console

The first step is to log in to the UDDI Service Console. Follow these steps

1. Open your web browser.
2. Type the following address in the address bar to open the TIBCO ActiveMatrix UDDI Service Console login page:

   `http://`*Host*`:`*Port*`/uddisc`
3. In the Username and Password fields, enter the username and password defined during the UDDI Server configuration (The default username/password is `admin/admin`) or any other username and password created in the Administration Console.
4. Click the **Log in** button to log in to Service Console.

For detailed information about how to log in to the UDDI Service Console, refer to Logging in to Service Console on page 12.

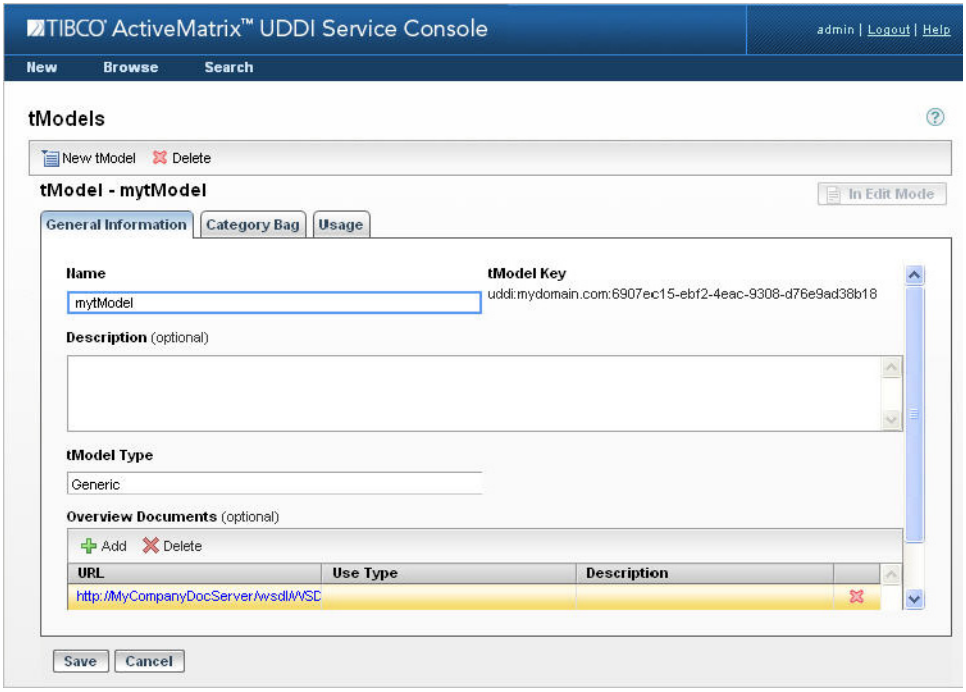
Creating a tModel Containing the WSDL Document

The WSDL document is used to describe how to access a service. It covers all UDDI operations that can be performed in the UDDI Service Console.

After logging in to the UDDI Service Console, you first need to create a tModel for the WDSL document. Follow these steps:

1. Select **New** > **tModel** on the menu to show the `tModel-New tModel` pane on the screen.
2. In the `tModel-New tModel` pane, enter information in the corresponding fields:
   - Enter **`mytModel`** as the tModel name in the Name field.
   - Click the **Add** button to add the URL of the WSDL document (for example, **`http://MyCompanyDocServer/wsdl/WSDLOfMyService.wsdl`**) in the Overview Document list box, as shown in Figure 3.

Figure 3 Entering a WSDL Document URL



3. click the **Save** button. The `tModel-mytModel` pane will appear and show information about `mytModel`.

For detailed information about how to create a tModel, refer to Creating a tModel on page 16.

Creating a Business

After creating the `mytModel` object, you need to create a business.

To create a business, follow these steps:

1. Select **New** > **Business** on the menu to show the `New Business` pane on the screen.
2. In the `Business - New Business` pane on the right, enter **`myBusiness`** as the business name in the Name field.
3. Click the **Save** button. The `myBusiness` pane will appear and show information about `MyBusiness` on the right.

For detailed information about how to create a business, refer to Creating a Business on page 20.

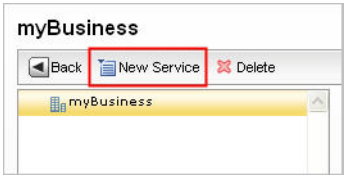
Creating a Service

After creating the `MyBusiness` object, you can create a service under it.

To create a service, follow these steps:

1. In the `myBusiness` pane, select the **myBusiness** object on the left, and then click the **New Service** button, as shown in Figure 4. The `Service-New Service` pane appears on the right.

Figure 4 New Service



2. Enter **`myService`** as the service name in the Name field.
3. Click the **Save** button. The `myService` information will appear on the right pane.

For detailed information about how to create a service, refer to Creating a Service for Your Business on page 23.

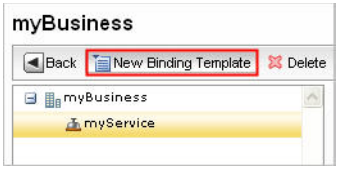
Creating a Binding Template

After creating `myService` under `myBusiness`, you can create a binding template under `myService`.

To create a binding template under `myService`, follow these steps:

1. In the `myBusiness` pane, select **myService** under the `myBusiness` object, and then click the **New Binding Template** button, as shown in Figure 5. The `Binding Template-New Binding` pane appears on the right.

Figure 5 New Binding Template



2. Enter a URL of the access point in the Access Point field, for example: **`http://MyProductionMachine/MyServiceRequest`**.
3. Click the **Save** button. The binding template information will appear the right pane.

For detailed information about how to create a binding template, refer to Creating a Binding Template for Your Service on page 26.

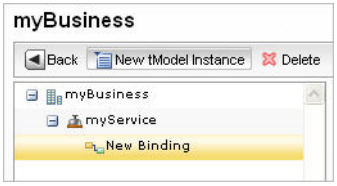
Creating a tModel Instance

After creating the binding template under the `myService` object, you need to create a tModel instance based on `mytModel` under the `myService` object.

To create a tModel instance, follow these steps:

1. In the `myBusiness` page, select a binding template, and then click the **New tModel Instance** button, as shown in Figure 6. The `tModel Instance-New tModelInstance` pane appears on the right.

Figure 6 New tModel Instance



2. Enter a tModel Key in the tModel Key field or click **Lookup** to search for your desired tModel key.
3. Click the **Save** button. The tModel instance information will appear on the right pane.

For detailed information about how to create a tModel instance, refer to Creating a tModel Instance on page 29.

Logging in to Service Console

You can log in to the UDDI Service Console as one of these four roles: read-only, subscribers, publishers, and administrators. These roles are described below.

- `readonly`: has inquiry permission which means you can invoke the inquiry API defined by the UDDI 3.0.2 spec.
- `subscribers`: has inquiry and subscription permissions.
- `publishers`: has inquiry and publication permissions.
- `administrators`: has all permissions that the above roles have and also includes permission to log in to Administration Console.

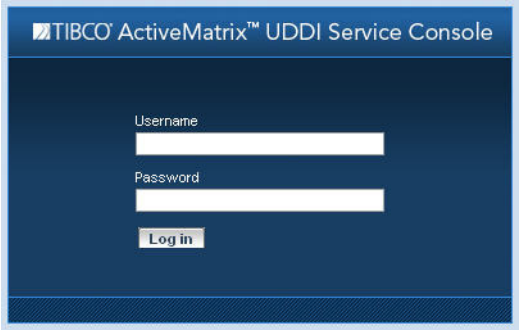
- If you have the *subscribers* role and need to publish subscriptions using asynchronous notification, you need to obtain the *publishers* role.
- If you have the *publishers* role and need to perform subscribing operations, you also need to obtain the *subscribers* role.
- Currently, user role control is not provided in UDDI Service Console.

For more information about user management, refer to *TIBCO ActiveMatrix Runtime UDDI Server Administrator's Guide.*

To perform TIBCO ActiveMatrix Runtime UDDI Server operations in UDDI Service Console, follow these steps to log in to UDDI Service Console first:

1. launch a web browser, enter `http://`*Host*`:`*Port*`/uddisc` (The URL for the installed default server is **`http://localhost:58080/uddisc`**) in the address bar, then press **Enter** to open the login page, as shown in Figure 7.

Figure 7 TIBCO ActiveMatrix UDDI Service Console Login Page



2. In the Username and Password fields, enter the username and password defined during the UDDI Server configuration (The default username/password is `admin/admin`) or any other username and password created in the Administration Console, and then click the **Log in** button.

   For more information about the Administration Console, refer to the Administration Console section in TIBCO ActiveMatrix Runtime UDDI Server Administrator's Guide.

   If the username or the password is invalid, you will see the following message shown on the login page:

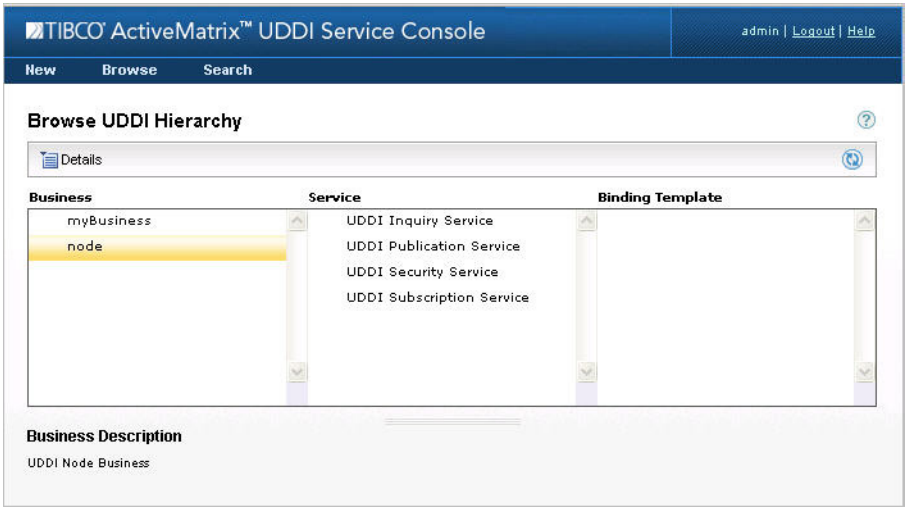
   ```
   The username or password is incorrect. Please try again.
   ```

- *Host* and *Port* in the URL used to access UDDI Service Console are set in the `srvconfig.xml` file. For more information, refer to *TIBCO ActiveMatrix Runtime UDDI Server Administrator's Guide*.
- If Websphere Application Server is used as the web container, the URL used to access UDDI Service Console needs to be in the format of `http://`*Host*`:`*Port*`/uddisc/index.html`.
- If UDDI Service Console is deployed on an HTTPS web server, the URL used to access UDDI Service Console needs to be in the format of `https://`*Host*`:`*Port*`/uddisc/`
- UDDI Service Console does not support anonymous login. Even on the server side and with anonymous inquiry enabled, you must provide a valid username and password to log in to UDDI Service Console.

The Browse UDDI Hierarchy Page

After you successfully log in to Service Console, the Browse UDDI Hierarchy page appears, as shown in Figure 8.

Figure 8 The Browse UDDI Hierarchy Page



The Browse UDDI Hierarchy page contains three columns: Business, Service, and Binding Template. It also has a Description section.

- The Business column lists the names of the business currently registered in TIBCO ActiveMatrix Runtime UDDI Server. When the page first appears, the column lists all the business objects maintained in the registry in alphabetical order.
- The Service column lists the services of the currently selected business.
- The Binding Template column lists the binding templates of the currently selected service. To display the binding templates of a service, you must select the service first. The binding templates are listed by their business keys, that is, in the order the binding templates are added to the service.
- The Description section displays the description of the object currently selected.

Viewing the Detailed Information of an Object

To view the detailed information of an object shown in the Browse UDDI Hierarchy page, follow these steps:

1. Select a desired object in the corresponding column.

Because the objects listed in the UDDI Hierarchy page are organized in a hierarchy, you must first select the root object and parent object to select a service object or a binding template object.

2. Click the **Details** button or double-click the selected object to open the object information page, as shown in Figure 9.

   The object information page has two panes and a tool bar:

   — The object hierarchy pane is on the left. It displays a four-level object tree that reflects the current object hierarchy. The root of the tree is the currently selected business or the business to which the currently selected service or binding template belongs to.

   — The object information pane is on the right. It displays the information about the currently selected object. It contains one or multiple tabs depending on the object you select.

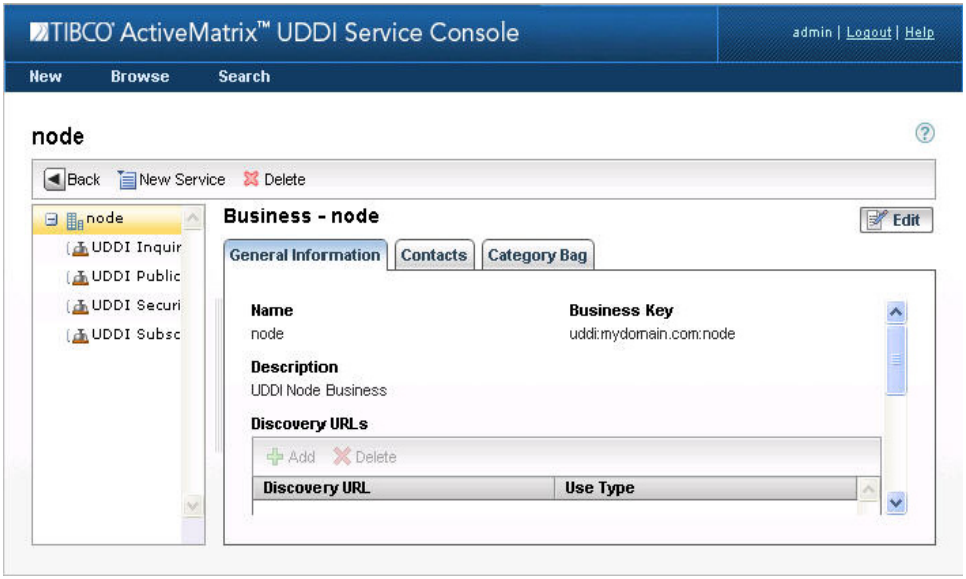
   — The tool bar is on the top. You can perform the following operations using the buttons on it.

   Click the **Back** button ( ) to return to the UDDI Hierarchy page.

   Click the **New** button ( ) to create a object.

   Click the **Delete** button ( ) to remove a selected object.

Figure 9 The Object Information Page



To terminate the current session, click **Log out** in the upper right corner in the Browse UDDI Hierarchy page.

tModel Operations

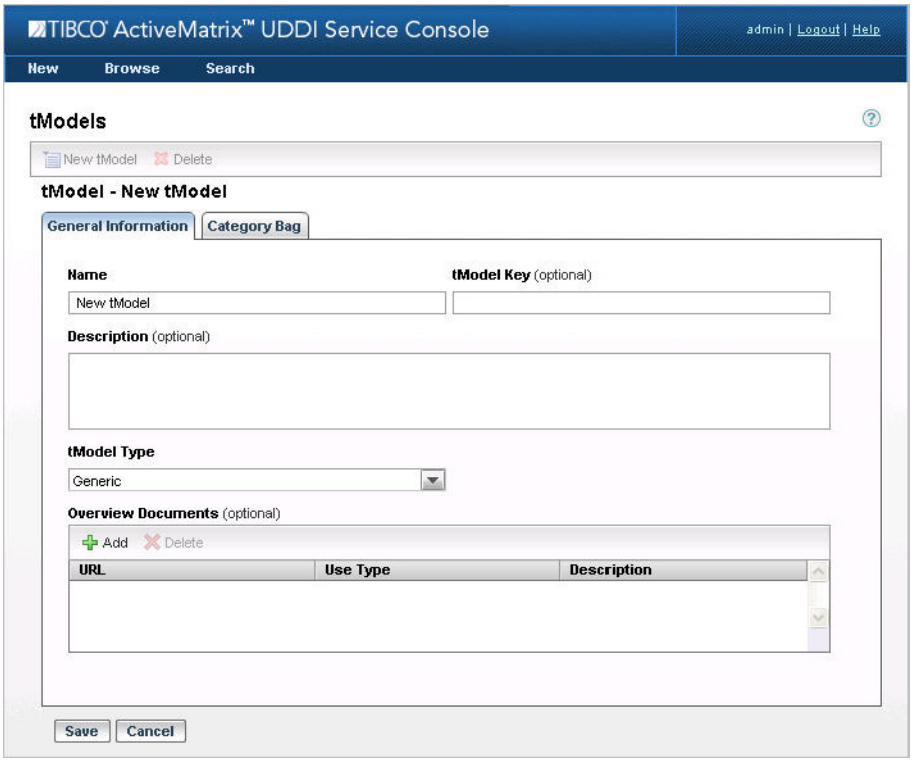
Technical Models, or tModels for short, are used in UDDI to represent unique concepts or constructs. This section introduces the basic operations performed on a tModel.

Creating a tModel

To create a tModel, follow these steps:

1. Select **New** > **tModel** from the UDDI Service Console menu to open the New tModel page, as shown in Figure 10.

Figure 10 The New tModel Page



2. Enter information in the following corresponding fields.

   In the General Information tab:

   — Name (required): Enter the tModel name in the Name field. The tModel name can contain characters, numbers, and spaces, but leading spaces and trailing spaces are not allowed.

   — tModel Key (optional): Enter the tModel key in this field. The format of the tModel Key is `uddi:`*yourServerDomainName*`:`*UniqueString*. For example, `uddi:mydomain.com:HRValidationTModel`

The tModels with their UDDI keys beginning with `uddi:uddi.org` are system canonical tModels that should never be deleted or modified.

Conversely, when creating a tModel, make sure the UDDI key provided does not begin with `uddi:uddi.org`.

In addition to these types of tModels, those with the following UDDI keys are reserved and cannot be deleted or modified.

- `uddi:amx:keygenerator`
- `uddi:297aaa47-2de3-4454-a04a-cf38e889d0c4`
- `uddi:70a80f61-77bc-4821-a5e2-2a406acc35dd`
- `uddi:db77450d-9fa8-45d4-a7bc-04411d14e384`
- `uddi:cd153257-086a-4237-b336-6bdcbdcc6634`

   — Description (optional): Enter a description of the tModel in the Description field.

   — tModel Type (required): Select a tModel Type from the tModel Type drop-down list.

   — Overview Documents (optional): Click the **Add** button to add a URL of the WSDL document.

   In the Category Bag tab, click the **Add** button to add categories referenced by the tModel.

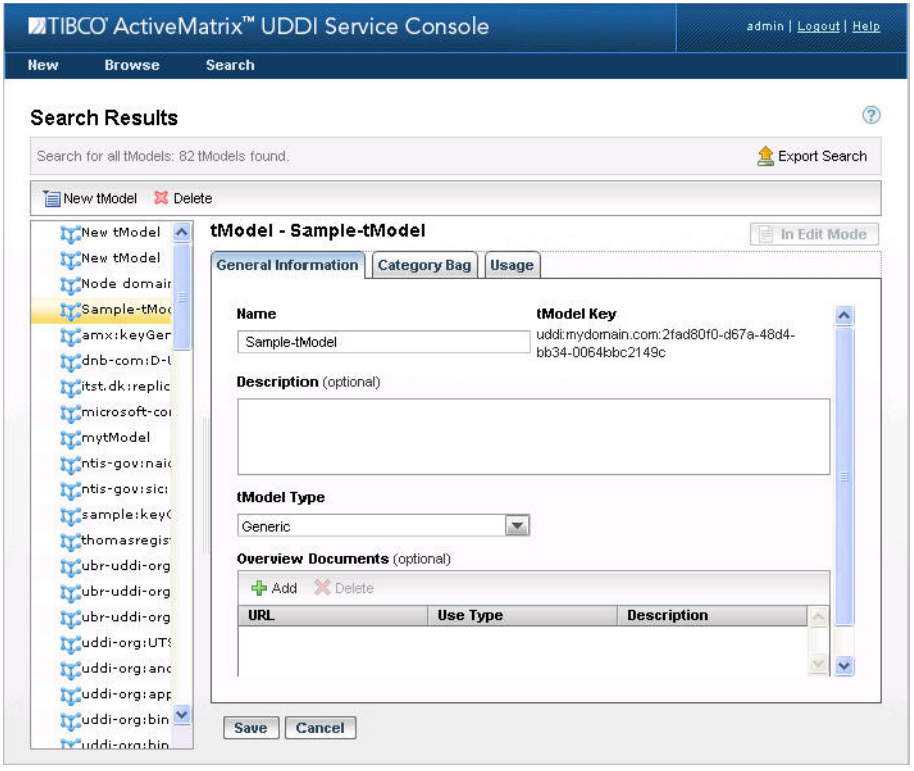
3. Click the **Save** button.

Editing a tModel

After creating a tModel, if you want to change its information, follow these steps:

1. Perform a search operation, and find the desired tModel in the object list pane of the Search Results page. For information on how to perform a search operation, see Searching for an Object on page 38.
2. Select the desired tModel and click the **Edit** button in the upper right corner. The Edit button changes to the In Edit Mode button. All the tModel settings in the object information pane become editable, as shown in Figure 11.

Figure 11 Editing a tModel



3. Make changes in the tabs and click the buttons as needed.
   - Click the **Save** button to save your changes and quit the edit mode.
   - Click the **Cancel** button to cancel the changes just made.

Removing a tModel

To remove a tModel, follow these steps:

1. Perform a search operation and find the desired tModel in the object list pane of the Search Results page. For information on how to perform a search operation, see Searching for an Object on page 38.
2. Select the tModel and then click the **Delete** button on the tool bar.
3. In the Delete Confirmation page, click the **Yes** button to delete the tModel.

Business Operations

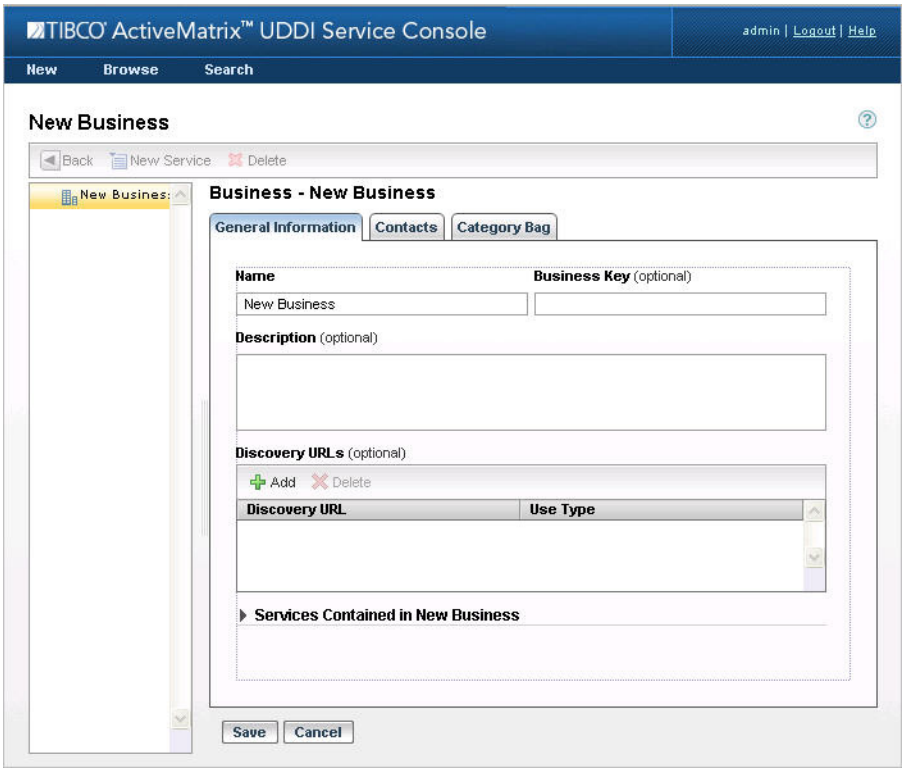
This section introduces the basic operations performed on a business.

Creating a Business

To create a business, follow these steps:

1. Select **New** > **Business** from the UDDI Service Console menu to open the New Business page, as shown in Figure 12.

Figure 12 The New Business Page



2. Enter information in the following corresponding fields.

In the General Information tab:

— Name (required): Enter the business name in the Name field. The business name can contain characters, numbers, and spaces, but leading spaces and trailing spaces are not allowed.

— Business Key (optional): Enter the business key in the Business Key field. The business key is unique for identifying the binding template. If you do not specify a business key here for your business, it will be generated automatically when you click the **Save** button. For more information about business keys, see Appendix A, UDDI Key Partition and Key Generator tModel.

— Description (optional): Enter a description of the business in the Description field.

— Discovery URLs (optional): Click the **Add** button to add the discovery URLs in the Discovery URLs list. A Discovery URL provides a link to additional technical or descriptive information about a business.

— Services Contained in New Business: Click the **New Service** button to add a service under the newly created business.

In the Contacts tab, click the **New** button to add contacts of the business.

In the Category Bag, click the **Add** button to add categories referenced by the business.

3. Click the **Save** button.

After you successfully create a business, it will be listed in the Business column of the Browse UDDI Hierarchy page.

Editing a Business

After creating a business, if you want to change its information, follow these steps:

1. Select the business you want to edit in the Browse UDDI Hierarchy page and click the **Details** button to display its information on the right pane.
2. Click the **Edit** button in the upper right corner. The Edit button changes to the In Edit Mode button. All the business settings in the object information pane become editable.
3. Make changes in the tabs and click the buttons as needed.
   — Click the **Save** button to save your changes and quit the edit mode.
   — Click the **Cancel** button to cancel the changes just made.

Removing a Business

To remove a business, follow these steps:

1. Select the business you want to remove from TIBCO ActiveMatrix Runtime UDDI Server in the Browse UDDI Hierarchy page and click the **Details** button to display the object information page.
2. Click the **Delete** button on the tool bar.
3. In the Delete Confirmation page, click the **Yes** button to delete the business.

Removing a business also removes all the services contained in the business.

Service Object Operations

This section introduces the basic operations performed on a service.

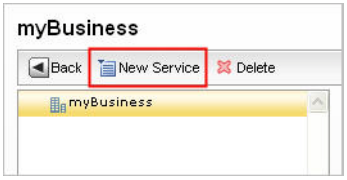
Creating a Service for Your Business

If you have created a business, you can create a service object for it.

To create a service object, follow these steps:

1. Select the business for which you want to create a service in the Browse UDDI Hierarchy page, and click the **Details** button in the upper left corner to open the object information page.
2. Click the **New Service** button on the tool bar to open the New Service page, as shown in Figure 13.

Figure 13 The New Service Page



3. Enter information in the following corresponding fields.

   In the General Information tab:

   — Name (required): Enter the service name in the Name field. The service name can contain characters, numbers, and spaces, but leading spaces and trailing spaces are not allowed.

   — Service Key (optional): Enter the service key in the Key field. The service key is unique for identifying the service. If you do not specify a service key here for your service, it will be generated automatically when you click the **Save** button. For more information about business keys, see Appendix A, UDDI Key Partition and Key Generator tModel.

   — Description (optional): Enter a description of the service in the Description field.

   — Business Key (read-only): List the reference business key.

   — Binding Templates Contained in New Service: Click the **New Binding Template** button to add a binding template under the newly created service.

   In the Category Bag, click the **Add** button to add categories referenced by the service.

4. Click the **Save** button.

After you successfully create a service, it will be listed in the Service column of the Browse UDDI Hierarchy page.

Editing a Service

After creating a service, if you want to change its information, follow these steps:

1. Select the service you want to edit in the Browse UDDI Hierarchy page and click the **Details** button to display its information on the right pane.
2. Click the **Edit** button in the upper right corner. The Edit button changes to the In Edit Mode button. All the service settings in the object information pane become editable.
3. Make changes in the tabs and click the buttons as needed.

   — Click the **Save** button to save your changes and quit the edit mode.

   — Click the **Cancel** button to cancel the changes just made.

Removing a Service

To remove a service, follow these steps:

1. Select the service you want to remove from TIBCO ActiveMatrix Runtime UDDI Server in the Browse UDDI Hierarchy page and click the **Details** button to display the object information page.
2. Click the **Delete** button on the tool bar.
3. In the Delete Confirmation page, click the **Yes** button to delete the service.

Removing a service also removes all the binding templates contained in the service.

Binding Template Object Operations

This section introduces the basic operations performed on a binding template.

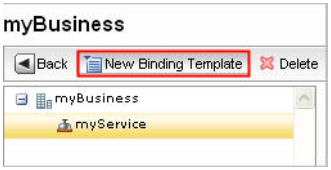
Creating a Binding Template for Your Service

If you have created a service, you can create a binding template for it.

To create a binding template for your service, follow these steps:

1. Select the service for which you want to create a binding template in the Browse UDDI Hierarchy page, and click the **Details** button on the tool bar to open the object information page.
2. Click the **New Binding Template** button on the tool bar to open the New Binding page, as shown in Figure 14.

Figure 14 The New Binding Template Page



3. Enter information in the following corresponding fields.

   In the General Information tab:

   — Binding Template Key (optional): Enter the binding template key in the field. The binding template key is unique for identifying the binding template. If you do not specify a binding template key here for your binding template, it will be generated automatically when you click the **Save** button. For more information about business keys, see Appendix A, UDDI Key Partition and Key Generator tModel.

   — Description (optional): Enter a description of the binding template in the Description field.

   — Service Key (read-only): List the reference service key.

   — Service Name (read-only): List the reference service name.

   — Access Point (required): Enter the access point in this field.

   — Access Point Type (optional): Enter the type of the access point in this field.

   — tModel Instances contained in Binding Template: Click the **New tModel Instance** button to add a tModel instance under the newly created binding template.

   In the Category Bag, click the **Add** button to add categories referenced by the binding template.

4. Click the **Save** button.

After you successfully create a binding template, it will appear in the object tree and in the Binding Template column of the Browse UDDI Hierarchy page.

Editing a Binding Template

After creating a binding template, if you want to change its information, follow these steps:

1. Select a binding template you want to edit in the Browse UDDI Hierarchy page. Click the **Details** button on the tool bar to display its information on the right pane, or expand the corresponding service in the object hierarchy pane to select the desired binding template.
2. Click the **Edit** button in the upper right corner. The Edit button changes to the In Edit Mode button. All the service settings in the object information pane become editable.

3. Make changes in the tabs and click the buttons as needed.
   - Click the **Save** button to save your changes and quit the edit mode.
   - Click the **Cancel** button to cancel the changes just made.

Removing a Binding Template

To remove a binding template, follow these steps:

1. Select a binding template you want to remove from TIBCO ActiveMatrix Runtime UDDI Server in the Browse UDDI Hierarchy page and click the **Details** button on the tool bar to display the object information page.
2. Click the **Delete** button on the tool bar.
3. In the Delete Confirmation page, click the **Yes** button to delete the binding template.

Removing a binding template also removes all the tModel instances contained in the binding template.

tModel Instance Operations

This section introduces the basic operations performed on a tModel instance.

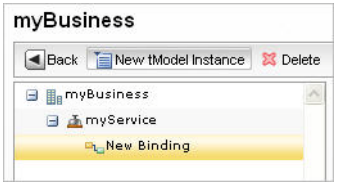
Creating a tModel Instance

If you have created a binding template, you can create a tModel instance for it.

To create a tModel instance for your binding template, follow these steps:

1. Select a binding template for which you want to create a tModel instance in the Browse UDDI Hierarchy page, and click the **Details** button on the tool bar to open the object information page.
2. Click the **New tModel Instance** button on the tool bar to open the New tModel Instance page on the right, as shown in Figure 15.

Figure 15 The New tModel Instance Dialog



3. Enter information in the following corresponding fields.

   In the General Information tab:

   — tModel Key (required): Enter an existing tModel in the tModel Key field. Click Lookup to check for available tModel keys and make sure the tModel key entered is valid.

   — Description (optional): Enter the description of tModel instance in the Description field.

   — Instance Parameters (optional): Enter the instance parameters in the Instance Parameters field.

   — Overview Documents (optional): Click the **Add** button to add the URL of the WSDL document.

4. Click the **Save** button.

After you successfully create a tModel instance, the tModel instance will appear in the object tree.

A tModel instance is an instance of a generic tModel that is associated with a binding template for categorization. Multiple tModel instances can be associated with the same binding template.

Editing a tModel Instance

After creating a tModel instance, if you want to change its information, follow these steps:

1. Select a tModel instance you want to edit in the object hierarchy pane and open its information pane on the right.
2. Click the **Edit** button in the upper right corner. The Edit button changes to the In Edit Mode button. All the tModel instance settings in the object information pane become editable.
3. Make changes in the tabs and click the buttons as needed.

   — Click the **Save** button to save your changes and quit the edit mode.

   — Click the **Cancel** button to cancel the changes just made.

Removing a tModel Instance

To remove a tModel instance, follow these steps:

1. Select a tModel instance you want to remove from TIBCO ActiveMatrix Runtime UDDI Server in the object hierarchy pane.
2. Click the **Delete** button on the tool bar.
3. In the Delete Confirmation page, click the **Yes** button to delete the tModel instance.

Category Operations

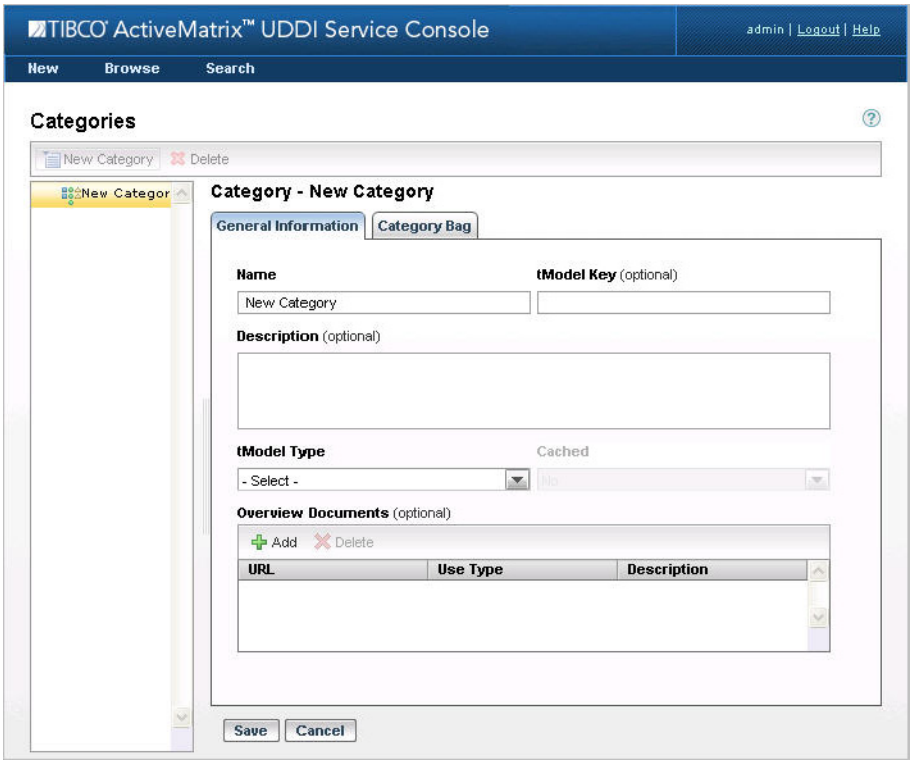
This section introduces the basic operations performed on a tModel instance.

Creating a Category

To create a category, follow these steps:

1. Select **New** > **Category** from the UDDI Service Console menu to open the New Category page, as shown in Figure 16.

Figure 16 The New Category Page



2. Enter information in the following corresponding fields.

In the General Information tab:

— Name (required): Enter the name of the category in the Name field. The category name can contain characters, numbers, and spaces, but leading spaces and trailing spaces are not allowed.

— tModel Key (optional): Enter the category key in the tModel Key field. This key is unique for identifying the category. It can contain characters, numbers, and special characters (such as hyphens).

— Description (optional): Enter a description of the category in the Description field.

— tModel Type (required): Select **Checked** or **Unchecked** as the category type in the tModel Type drop-down list.

— Cached: If you select **Checked** as the category type, the Cached drop-down list is available, and you need to select **Yes** or **No** from it.

— Overview Documents (optional): Click the **Add** button to add a URL of the WSDL document.

In the Category Bag tab, click the **Add** button to add categories referenced by the tModel.

A tModel instance is created from a tModel category. This means that you need to create the corresponding tModel category before creating the tModel instance. For more information about using categories, see Chapter 7, Using Category Systems, on page 105.

Editing a Category

After creating a category, if you want to change its information, follow these steps:

1. select **Browse** > **Categories** from the UDDI Service Console menu to open the Categories page.
2. Select a category you want to edit and click the **Edit** button in the upper right corner. The Edit button changes to the In Edit Mode button. All the category settings in the object information pane become editable.
3. Make changes in the tabs and click the buttons as needed.
   — Click the **Save** button to save your changes and quit the edit mode.
   — Click the **Cancel** button to cancel the changes just made.

Removing a Category

To remove a category, follow these steps:

1. select **Browse** > **Categories** from the UDDI Service Console menu to open the Categories page.
2. Select a category you want to delete in the left pane and click the **Delete** button on the tool bar.
3. In the Delete Confirmation page, click the **Yes** button to delete the category.

Subscription Operations

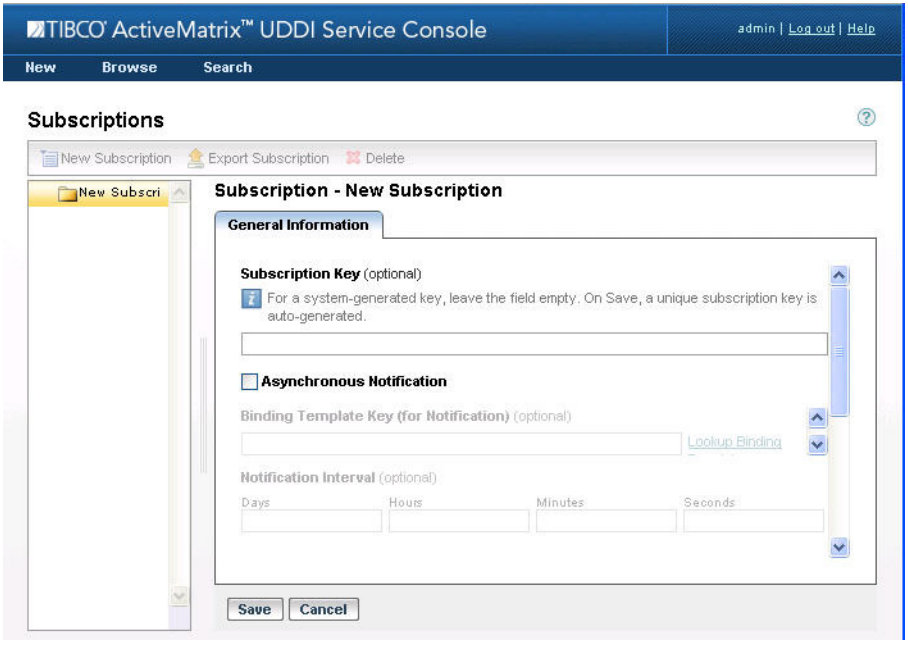
Subscription is used to monitor changes in a UDDI registry. This section introduces the basic operations performed on a tModel instance.

Creating a Subscription

To create a subscription, follow these steps:

1. Select **New** > **Subscription** from the UDDI Service Console menu to open the Subscriptions page, as shown in Figure 17. The New Subscription information is shown on the right pane.

Figure 17 The New Subscription Dialog



2. Enter information in the following corresponding fields.

In the General Information tab:

— Subscription Key (optional): Enter the subscription key in the Subscription Key field. The subscription key is unique for identifying the subscription. If you do not specify a key here for your subscription, it will be generated automatically when you click the **Save** button.

— Asynchronous Notification (optional): Check or uncheck the Asynchronous Notification checkbox to indicate whether you want to set asynchronous notification when you create a subscription. If you check the checkbox, the following two fields will appear:

Binding Template Key (for Notification): look up a binding template that includes an email receiver's information.

Notification Interval: Set the notification interval including days, hours, minutes, and seconds.

— Expiry Date and Time (optional): Set the expiry date and time. Normally, a subscription is valid for a specific period, so you need to specify the expiration date and time. Once a subscription is created and saved, the UDDI server will record the changes of the entities matching the filter criteria in the specified time period and will stop recording when the subscription period expires.

— Return Brief Messages (optional): Select TRUE or FALSE from the drop-down list to indicate whether you want to return brief messages.

— Subscription Filter (required): You need to specify the entities you want to monitor by setting the subscription filter, through which only the entities matching the filter criteria are monitored.

Click **Modify** to open the Modify Subscription Filter page. You can modify the subscription filter under the By Name or Key tab or the Categories tab.

3. Click the **Save** button.

Exporting a Subscription

To export a subscription, follow these steps:

1. select **Browse** > **Subscriptions** from the UDDI Service Console menu to open the Subscriptions page.
2. Select a subscription in the left pane and click the **Export Subscription** button on the tool bar. you can export the information about the currently selected subscription to an XML file.

Editing a Subscription

After creating a subscription, if you want to change its information, follow these steps:

1. select **Browse** > **Subscriptions** from the UDDI Service Console menu to open the Subscriptions page.
2. Select a subscription you want to edit and click the **Edit** button in the upper right corner. The Edit button changes to the In Edit Mode button. All the subscription settings in the object information pane become editable.
3. Make changes in the tabs and click the buttons as needed. To modify the subscription filter, click the **Subscription Filter Details** link.
   - Click the **Save** button to save your changes and quit the edit mode.
   - Click the **Cancel** button to cancel the changes just made.

Removing a Subscription

To remove a category, follow these steps:

1. Select **Browse** > **Subscriptions** from the UDDI Service Console menu to open the Subscriptions page.
2. Select a subscription you want to remove in the left pane and click the **Delete** button on the tool bar.
3. In the Delete Confirmation page, click the **Yes** button to delete the subscription.

Searching for an Object

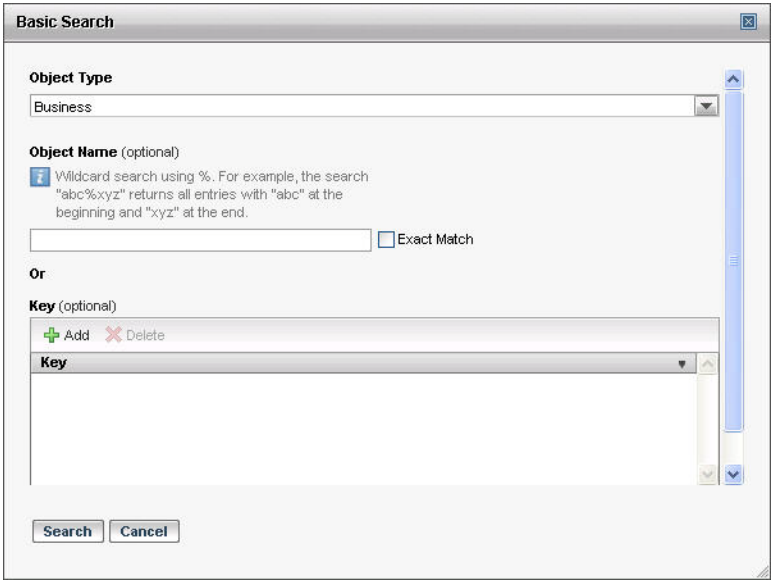
To work with an object, you need to locate it first. UDDI Service Console provides two search functions: the basic search function and the search by category function.

Performing a Basic Search

To perform a basic search, follow these steps:

1. Select **Search** > **Basic** from the UDDI Service Console menu to open the Basic Search dialog, as shown in Figure 18.

Figure 18 The Basic Search Dialog



2. Enter information in the following corresponding fields.
   - Object Type: Select the object type in the Object Type drop-down list. You can select one of these object types: **Business**, **Service**, **Binding Template**, and **tModel**.
   - Object Name: Enter the object name in the Object Name field.

You can use % to perform the wildcard search if the **Exact Match** checkbox is unchecked.

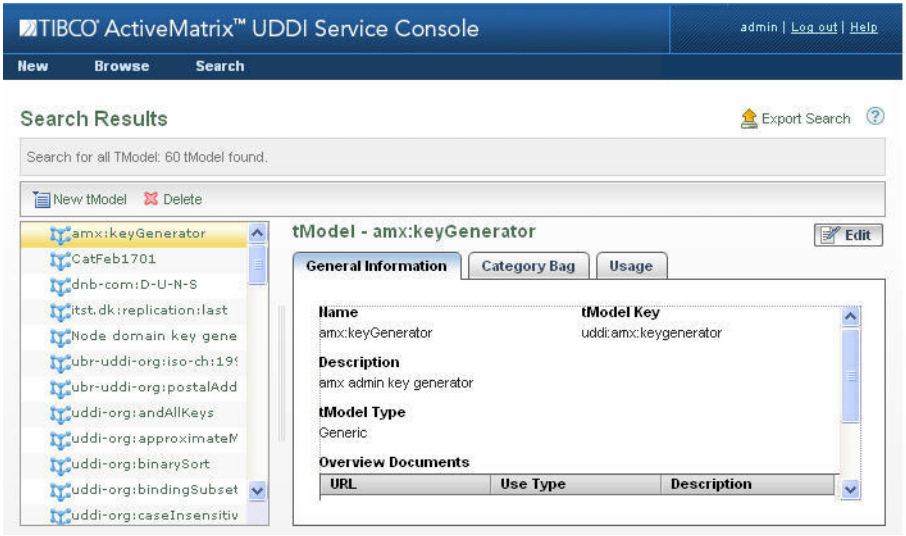
- The search `abc%` returns all entries with `abc` at the beginning.
- The search `%xyz` returns all entries with `xyz` at the end.
- The search `abc%xyz` returns all entries with `abc` at the beginning and `xyz` at the end.
- The search `abc` returns all entries with `abc` in anywhere.

If you check the **Exact Match** checkbox, an exact search will be performed.

   - Key: Click the **Add** button to add the keys in the Key list.

3. Click the **Search** button to open the Search Results page, as shown in Figure 19 (for tModel objects).

Figure 19 The Search Results Page (1)



4. Click the **Export Search** button in the upper right corner, you can export the searching criteria as an XML file named `exportData.xml`.

The `exportData.xml` file can be used as a data file when performing the `find_XXX` or `get_XXX` tasks in the Command Line Interface.

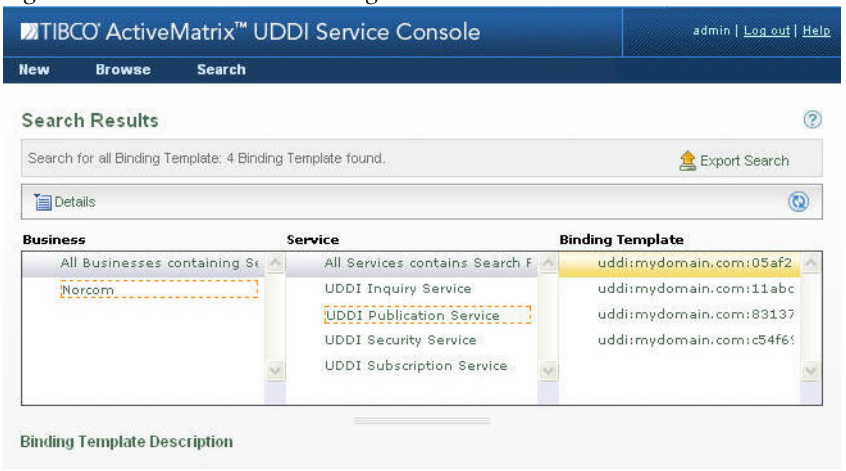
For more information about the `find_XXX` or `get_XXX` tasks, refer to Find Tasks on page 52 and Get Tasks on page 59.

Search Results

The search result varies with the search criteria, as described below.

- If only one object matches the criteria, the search result contains only the object.
- If you search for tModels and multiple tModels match the criteria, the tModels will be listed in the left pane of the Search Result page. The right pane displays the information about the tModel object currently selected in the left pane.
- If you search for Business objects, Service objects, or binding templates and multiple objects match the criteria, the Search Results page appears (as shown in Figure 20) after you click the **Search** button. In addition to displaying the object you are searching for, this page also marks the parent objects of the currently selected object by enclosing the parent objects in red dashed frames.

Figure 20 The Search Results Page (2)



After you perform a basic search, the settings in the Search page remain until you change them in another basic search.

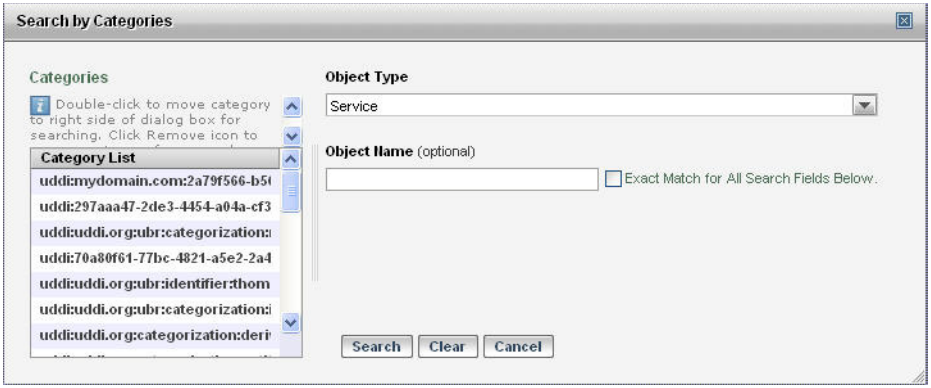
Searching for an Object by Categories

You can search for an object by object type, object name, and category name and value.

To search by categories, follow these steps:

1. Select **Search** > **by Categories** from the UDDI Service Console menu to open the Search by Categories dialog, as shown in Figure 21.

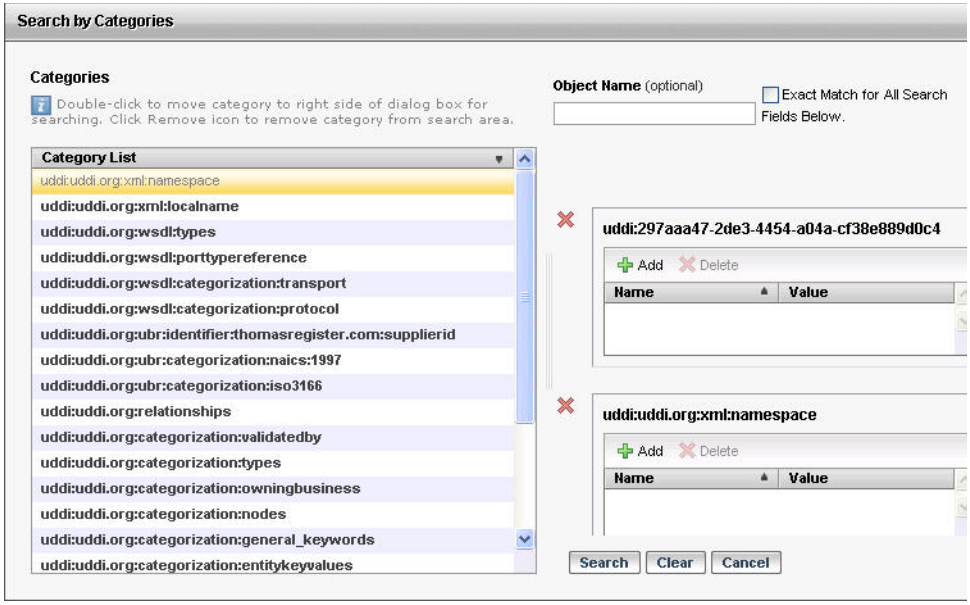
Figure 21 The Search by Category Dialog



In the Search by Categories page, the list on the left of the page lists the existing categories in the order the server returns them. The Object Type drop-down list and the Object Name field are on the right side of the page.

2. In the Category List on the left pane, double-click a category you want to use in the search to open the Search by Categories dialog, as shown in Figure 22. A table for the category has been added to the dialog. You can refine the search criteria by adding name-value pairs in the cells of the table.

Figure 22 Specifying a Category for Searching for an Object



Provide name-value pairs in the cells of the tables and click the **Add** button to add the name-value pair.

You can add multiple name-value pairs for a category in the same way. To remove an existing name-value pair, click the Delete button in the corresponding row. If no name-value pair is set, the search will be performed with the category ignored.

3. Repeat step 2 to add other categories as needed for searching. To disable a category from being used in the search, click the **Delete** button in the top left corner of the corresponding table. Note that if multiple categories are specified, only the objects that match all the criteria specified for the categories simultaneously can be the search result.
4. Specify an exact search by selecting the **Exact Match for All Search Fields Below** check box.
5. Click the **Search** button to open the Search Results page, as shown in Figure 19 (for tModel objects) and Figure 20 (for business objects, service objects, and binding template objects).

6. Click the **Export Search** button in the top right corner of the page to export the search criteria to an XML file named `exportData.xml`.

The `exportData.xml` file can be used as a data file when performing the `find_XXX` or `get_XXX` tasks in the Command Line Interface.

For more information about the `find_XXX` or `get_XXX` tasks, refer to Find Tasks on page 52 and Get Tasks on page 59.

For information about the search results, refer to Search Results on page 40.

Chapter 3 Command Line Interface

This chapter describes how to execute Ant-based tasks in the Command Line Interface (CLI).

Topics

Overview

TIBCO ActiveMatrix Runtime UDDI Server allows you to execute the following types of Ant-based tasks in the CLI.

- Connecting to the UDDI Server, page 47
- Find Tasks, page 52
- Get Tasks, page 59
- Save Tasks, page 63
- Delete Tasks, page 70
- Subscription Tasks, page 72
- Security Tasks, page 78

The following two methods can be used to execute these Ant-based tasks:

— Run **`ant -f uddiant.xml`** under the following directory:

TIBCO_HOME`/RuntimeUDDIServer/3.1/scripts/client`

— Run **`uddiant`** under the following directory:

TIBCO_HOME`/RuntimeUDDIServer/3.1/bin`

`Apache-ant-1.7.1` and `JRE 1.6.0` must be pre-installed and set into the system path if you want to work with TIBCO ActiveMatrix Runtime UDDI Server using `Ant-based` commands.

An Ant-based task can be executed in either the interactive mode or the silent mode.

- To execute an Ant-based task in the interactive mode, you need to provide the arguments needed for executing the task in the CLI when prompted.
- To execute an Ant-based task in the silent mode, you need to provide all the arguments through the `-D` parameter in the CLI along with the command. In the silent mode, Ant-based tasks are executed without prompting for any arguments because all arguments are provided in the command line including the information about the data files.

Connecting to the UDDI Server

To execute an Ant-based task, you must first connect to the UDDI server. You can provide connection information to the UDDI server using one of the following methods:

- Using the default properties file

  By default, an Ant-based task uses the URL properties in the `uddi.properties` file to connect to a UDDI server. You can modify the URL properties in the `uddi.properties` file to have the Ant-based tasks connect to the desired UDDI server.

  For detailed information about the `uddi.properties` file, refer to the UDDI Property File section.

- Providing the information in CLI through `-D` options. For example:

  `delete_service -DserviceKey=`*`ServiceKey`* `-Dinquiry_url=`*`INQUIRY_URL`*
  `-Dpublish_url=`*`PUBLISH_URL`* `-Dsecurity_url=`*`SECURITY_URL`*
  `-DuserName=`*`UserName`* `-Dpassword=`*`Password`*

To assign a value to a property through a `-D` option, you need to enclose the value in double quotation marks if the value contains spaces. For example, `-Dname="my value"`.

- Provide scripts for connect and disconnect tasks, as listed in Security Tasks on page 78.

UDDI Property File

The common properties required by all the Ant-based tasks are stored in the `uddi.properties` file. Each time you execute an Ant-based task, the task reads the properties stored in the file if you do not specify your own properties in command line through `-D` options.

By default, the `uddi.properties` file is stored in the following directory: *TIBCO_HOME*`/RuntimeUDDIServer/3.1/scripts/client`.

Normally, a `uddi.properties` file contains the following sections:

- URL Properties contain URL properties required by different types of Ant-based tasks.
- User confidential information contains the username and password used to access the server. You can provide an encrypted password for the `password` property.

- HTTPS Properties contain properties used to access HTTPS servers.
- Proxy Properties contain properties used to access proxies.

A `uddi.properties` file contains the following information:

```
inquiry_url = http://Host:Port/uddi/services/inquiry
publish_url = http://Host:Port/uddi/services/publication
security_url= http://Host:Port/uddi/services/security
subscription_url = http://Host:Port/uddi/services/subscription
admin_url = http://Host:Port/uddi/services/admin

userName = admin
password = #!OQ5liHOGFpZSuBG3qU0tP3WDkGaIDazm

trustStoreFolder=

keystorePath=
keystoreType=
keystorePassword=

proxyHost=
proxyPort=
proxyUsername=
proxyPassword=
```

where, *Host* is the name or the IP address of the host where TIBCO ActiveMatrix Runtime UDDI Server resides, and *Port* is the port number of the port through which UDDI services are provided.

If you want to inquiry anonymously, enable the anonymous inquiry in TIBCO ActiveMatrix Runtime UDDI Server and remove the username and password fields, or set `username`=*emptyString*, `password`=*emptyString* in the uddi.properties file.

URL Properties

Executing an Ant-based task requires specific URL properties. The URL properties required vary with Ant-based task types. Currently, the following five URL properties are used: inquiry_URL, publish_URL, security_URL, subscription_URL, and admin_URL. By default, the URL properties specified in the `uddi.properties` file are used. You can also specify your own URL properties using `-D` parameters in CLI when executing an Ant-based task, as shown below.

`uddiant save_busdiness -D`*PropertyName*=*PropertyValue*

Table 3, Ant-based Tasks and the Corresponding Required URL lists Ant-based tasks and the corresponding required URL properties.

Table 3 Ant-based Tasks and the Corresponding Required URL

Tasks	Required URL	Remarks
Find tasks	inquiry_URL, security_URL	With anonymous inquiry disabled, all the Ant-based tasks require the security_URL property. With anonymous inquiry enabled, the security_URL is unnecessary to find tasks and get tasks. Moreover, if only the inquiry_URL is used, the username and password properties can be left empty; if other URLs are used, the username and password are necessary. The read-only permission is required to access an inquiry_URL or a publication_URL. You need to be a publisher to access a publication_URL; you need to be a subscriber to access a subscription_URL. To create a subscription that adopts the asynchronous notification mechanism, both the publication_URL and the subscription_URL are required.
Get tasks	inquiry_URL, security_URL	
Save tasks	inquiry_URL, publication_URL, security_URL	
Delete tasks	inquiry_URL, publication_URL, security_URL	
Subscription tasks	subscription_URL, security_URL	

In order for TIBCO Business Studio, TIBCO ActiveMatrix Administrator, and TIBCO ActiveMatrix BusinessWorks to work with TIBCO ActiveMatrix Runtime UDDI Server, anonymous inquiry must be enabled in TIBCO ActiveMatrix Runtime UDDI Server. You can enable anonymous inquiry in TIBCO Administrator Console.

User confidential information

When connecting to the UDDI server, you are required to provide a username and the corresponding password. To avoid your password from being disclosed, you can encrypt your password by adding `#!` or `#!!` at the beginning of the password entered.

The strings `#!` and `#!!` encrypt passwords in different ways. A password prefixed by `#!` is encrypted using a fixed symmetric key. In this case, the password is valid on any machines. A password prefixed by `#!!` is encrypted using a key generated by the local machine. In this case, the password is valid only on the local machine.

Note that the string `#!` and `#!!` are reserved by TIBCO for password encrypting. Do not use them as the beginning of any clear text passwords.

For more information on encrypting a password, see Encrypting a Password for the UDDI Server in *TIBCO ActiveMatrix Runtime UDDI Server Administrator's Guide*.

HTTPS Properties

HTTPS uses digital certificates to control access to servers. To access an HTTPS server, you need to first import the corresponding digital certificates either as individual certificate files or to a trusted keystore file.

HTTPS properties allow Ant-based tasks to access HTTPS web servers. They provide the information about the digital certificates to be used and are required when the URLs assigned to the URL properties point to HTTPS servers. The HTTPS properties are described as follows.

- `trustStoreFolder`, the folder where the digital certificate files are located. If you import the digital certificates as individual certificate files, provide the path of the certificate files for this property.
- `keystorePath`, the path of the trusted keystore. If you import the digital certificates to a trusted keystore file, provide the path of the keystore file for this property.
- `keystoreType`, the type of the keystore. Keystore type is specified when a keystore is created. This property is required if you import the digital certificate to a trusted keystore file. The two keystore types commonly used in TIBCO products are JKS and PKCS12. For related information, see Appendix A: Standard Names in *Java Cryptography Architecture API Specification & Reference* at http://java.sun.com/javase/6/docs/technotes/guides/security/crypto/CryptoSpec.html.
- `keystorePassword`, the password used to access the trusted keystore. This property is required if you import the digital certificate to a trusted keystore file.

For TIBCO ActiveMatrix Runtime UDDI Server, only one-way certificate verification is supported. That is, only the client verifies the server's certificate. The server, however, does not verify the client's certificate. The client can be Ant-based tasks, HTTP GET requests, and Service Console requests.

Proxy Properties

Proxies are usually deployed for servers to implement additional control functions, such as monitoring and filtering the data and requests between servers and clients. For TIBCO ActiveMatrix Runtime UDDI Server, a proxy is usually used if the port assigned to it cannot be exposed externally. If a proxy is deployed and you want Ant-based tasks to access the server through the proxy, you need to set the proxy properties properly.

The proxy properties are described as follows.

- `proxyHost`, the IP address of the host operating as the proxy.
- `proxyPort`, the port on the proxy for HTTP requests.
- `proxyUsername`, the username used to access the proxy.
- `proxyPassword`, the password used to access the proxy.

- Proxies can be deployed only when HTTP is used to access the servers.
- Leave the proxy properties empty if you do not want to use a proxy to access the server.

Find Tasks

Execute one of the following commands in CLI to execute a find task in the interactive mode:

— ant -f `uddiant.xml find_`*sampleActivity*

— `uddiant find_`*sampleActivity*

Each find task prompts whether to use a data file. The following prompts will appear in CLI.

- `Enter the data from the data file (y, n)?`

  Press **Y** and then press **Enter** to use a data file. Otherwise, press **N**.

  If you press **Y** and then press **Enter**, the following prompts will appear.

- `Enter the data file (required).`

  Enter the absolute path and the name of the data file.

- `Enter the output file location in the relative or absolute path`

  To store the output information of the task in a file, enter the absolute path or the relative path and the name of the file.

  You can also display the output information in CLI by pressing **Enter**.

  If you specify to store the output information in a file, the following prompt may appear.

- `Overwrite the output file? ([true], false)`

  Specify the way to handle the specified output file. Selecting **`true`** will overwrite the file. Selecting **`false`** will append the output information to the file. The default is **`true`**.

If you press **N** and then press **Enter** in response to the `Enter the data from the data file (y, n)` prompt, the find task will prompt for the name of the related entity and a qualifier to be used while searching.

There are two options for the qualifier: `exactMatch` and `approximateMatch`.

- `exactMatch`: searches for the entities whose names match the input entity name exactly.

- `approximateMatch`: searches for the entities whose names match the input entity name approximately. If you select this qualifier, you can use the following wildcards.
  - — `%`: stands for multiple characters.
  - — `_`: stands for a single character.

  For example, to search services whose names start with `UDDI`, set the `serviceName` as `UDDI%` in the requested service name input.

The case-sensitive mode of the database determines the default case-sensitive mode of the find_business tasks, find_service tasks, and find_tModel tasks. That is, if the database is case-sensitive, the three types of tasks will be executed in the case-sensitive way by default; if the database is case-insensitive, the three types of tasks will be executed in the case-insensitive way by default.

With the database being case-sensitive, you can have the three types of tasks executed in the case-insensitive way by adding the following qualifier in the data file:

```
<findQualifiers>
    <findQualifier>caseInsensitiveMatch</findQualifier>
</findQualifiers>.
```

With the database being case-insensitive, you can have the three types of tasks executed in the case-sensitive way by adding the following qualifier in the data file:

```
<findQualifiers>
    <findQualifier>caseSensitiveMatch</findQualifier>
</findQualifiers>.
```

find_business

Execute one of the following commands in CLI to execute a find_business task in the interactive mode:

- — ant -f `uddiant.xml find_business`
- — `uddiant find_business`

If you press **N** and then press **Enter** in response to the `Enter the data from the data file (y, n)` prompt, the following prompts will appear.

- `Enter the Business Name(required)`

  Enter the name of the business.

- `Enter the Qualifier (exactMatch, [approximateMatch])`

Select **approximateMatch** or **exactMatch** as needed. The default is approximateMatch.

- `Enter the output file location in the relative or absolute path`

  To store the output information in a file, enter the relative or absolute path and the name of the file.

  You can also display the output information in CLI by pressing **Enter**.

- `Enter the number of results you want in the output file: [5]`

  Enter the number of result entries you want the task to return. The default is **5**.

  If you specify to store the output information in a file, the following may appear.

- `Overwrite the output file? ([true], false)`

  Specify the way to handle the specified output file. Selecting **true** will overwrite the file. Selecting **false** will append the output information to the file. The default is **true**.

You can also execute the task in silent mode by executing one of the following commands.

— ant -f `uddiant.xml find_business` -Ddatafromfile=[n|y]
`-DbusinessName=`*BusinessName*
`-Dqualifier=[exactMatch|approximateMatch]`
`-Dfile=`*OutputFilePath/OutputFileName* `-Doverwrite=[true|false]`
`-Dlimit=`*Integer*

— `uddiant find_business` -Ddatafromfile=[n|y]
`-DbusinessName=`*BusinessName*
`-Dqualifier=[exactMatch|approximateMatch]`
`-Dfile=`*OutputFilePath/OutputFileName* `-Doverwrite=[true|false]`
`-Dlimit=`*Integer*

For information about the structure of the data files used by find_business tasks, see Section 5.1.10 find_business in *UDDI Version 3.0.2* at http://www.uddi.org/pubs/uddi_v3.htm#_Toc85908086.

A sample data file `find_business.xml` is provided in the `TIBCO_HOME/RuntimeUDDIServer/3.1/sample/data` directory.

find_service

Execute one of the following commands in CLI to execute a find_service task in the interactive mode:

— ant -f `uddiant.xml find_service`

— `uddiant find_service`

If you press **N** and then press **Enter** in response to the `Enter the data from the data file (y, n)` prompt, the following prompts will appear.

- `Enter the Service Name(required)`

  Enter the name of the business service.

- `Enter the Qualifier (exactMatch, [approximateMatch])`

  Select **`approximateMatch`** or **`exactMatch`** as needed. The default is `approximateMatch`.

- `Enter the output file location in the relative or absolute path`

  To store the output information in a file, enter the relative or absolute path and the name of the file.

  You can also display the output information in CLI by pressing **Enter**.

- `Enter the number of results you want in the output file: [5]`

  Enter the number of result entries you want the task to return. The default is **`5`**.

  If you specify to store the output information in a file, the following may appear.

- `Overwrite the output file? ([true], false)`

  Specify the way to handle the specified output file. Selecting **`true`** will overwrite the file. Selecting **`false`** will append the output information to the file. The default is **`true`**.

You can also execute the task in silent mode by executing one of the following commands.

— ant -f `uddiant.xml find_service` -Ddatafromfile=[n|y]
`-DserviceName=`*`ServiceName`*
`-Dqualifier=[exactMatch|approximateMatch]`
`-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`
`-Dlimit=`*`Integer`*

— `uddiant find_service` -Ddatafromfile=[n|y]
`-DserviceName=`*`ServiceName`*
`-Dqualifier=[exactMatch|approximateMatch]`

```
-Dfile=OutputFilePath/OutputFileName -Doverwrite=[true|false]
-Dlimit=Integer
```

For information about the structure of the data files used by find_service tasks, see Section 5.1.12 find_service in *UDDI Version 3.0.2* at http://www.uddi.org/pubs/uddi_v3.htm#_Toc85908088.

A sample data file `find_service.xml` is provided in the `TIBCO_HOME/RuntimeUDDIServer/3.1/sample/data` directory.

find_tModel

Execute one of the following commands in CLI to execute a find_tModel task in the interactive mode:

— `ant -f uddiant.xml find_tModel`

— uddiant `find_tModel`

If you press **N** and then press **Enter** in response to the `Enter the data from the data file (y, n)` prompt, the following prompts will appear.

- `Enter the Tmodel Name(required)`

  Enter the name of the tModel.

- `Enter the Qualifier (exactMatch, [approximateMatch])`

  Select **`approximateMatch`** or **`exactMatch`** as needed. The default is `approximateMatch`.

- `Enter the output file location in the relative or absolute path`

  To store the output information in a file, enter the relative or absolute path and the name of the file.

  You can also display the output information in CLI by pressing **Enter**.

- `Enter the number of results you want in the output file: [5]`

  Enter the number of result entries you want the task to return. The default is **`5`**.

  If you specify to store the output information in a file, the following may appear.

- `Overwrite the output file? ([true], false)`

  Specify the way to handle the specified output file. Selecting **`true`** will overwrite the file. Selecting **`false`** will append the output information to the file. The default is **`true`**.

You can also execute the task in the silent mode by executing one of the following commands.

— `ant -f uddiant.xml find_tModel` -Ddatafromfile=[n|y] `-DtmodelName=`*tModelName* `-Dqualifier=[exactMatch|approximateMatch]` `-Dfile=`*OutputFilePath/OutputFileName* `-Doverwrite=[true|false]` `-Dlimit=`*Integer*

— uddiant `find_tModel` -Ddatafromfile=[n|y] `-DtmodelName=`*tModelName* `-Dqualifier=[exactMatch|approximateMatch]` `-Dfile=`*OutputFilePath/OutputFileName* `-Doverwrite=[true|false]` `-Dlimit=`*Integer*

For information about the structure of the data files used by find_tModel tasks, see Section 5.1.13 find_tModel in *UDDI Version 3.0.2* at http://www.uddi.org/pubs/uddi_v3.htm#_Toc85908089.

A sample data file `find_tModel.xml` is provided in the `TIBCO_HOME/RuntimeUDDIServer/3.1/sample/data` directory.

find_binding

Execute one of the following commands in CLI to execute a find_binding task in the interactive mode:

— `ant -f uddiant.xml find_binding`

— uddiant `find_binding`

If you press **N** and then press **Enter** in response to the `Enter the data from the data file (y, n)` prompt, the following prompts will appear.

- `Enter the Business Service key(required)`

  Enter the service key of the binding templates you want to search.

- `Enter the output file location in the relative or absolute path`

  To store the output information in a file, enter the relative or absolute path and the name of the file.

  You can also display the output information in CLI by pressing **Enter**.

- `Enter the number of results you want in the output file: [5]`

  Enter the number of result entries you want the task to return. The default is **5**.

  If you specify to store the output information in a file, the following may appear.

- `Overwrite the output file? ([true], false)`

  Specify the way to handle the specified output file. Selecting **`true`** will overwrite the file. Selecting **`false`** will append the output information to the file. The default is **`true`**.

You can also execute the task in silent mode by executing one of the following commands.

— `ant -f uddiant.xml find_binding` -Ddatafromfile=[n|y] `-DserviceKey=`*ServiceKey* `-Dfile=`*OutputFilePath/OutputFileName* `-Doverwrite=[true|false] -Dlimit=`*Integer*

— uddiant `find_binding` -Ddatafromfile=[n|y] `-DserviceKey=`*ServiceKey* `-Dfile=`*OutputFilePath/OutputFileName* `-Doverwrite=[true|false] -Dlimit=`*Integer*

For information about the structure of the data files used by find_binding tasks, see Section 5.1.9 find_binding in *UDDI Version 3.0.2* at http://www.uddi.org/pubs/uddi_v3.htm#_Toc85908085.

A sample data file `find_binding.xml` is provided in the `TIBCO_HOME/RuntimeUDDIServer/3.1/sample/data` directory.

Get Tasks

Each get task prompts for the UDDI key of the desired entity and returns the entity in the output file.

- The UDDI key entered for a get task must be a specific value. The character % is not allowed.
- You can use the output of a get task as the input of a save task.

get_bindingDetail

Execute one of the following commands in CLI to execute a get_bindingDetail task in the interactive mode:

— `ant -f uddiant.xml get_bindingDetail`

— uddiant `get_bindingDetail`

The following prompts will appear.

- `Enter the Binding Key(required)`

  Enter the UDDI key of the desired binding template.

- `Enter the output file location in the relative or absolute path`

  To store the output information in a file, enter the relative or absolute path and the name of the file.

  You can also display the output information in CLI by pressing **Enter**.

  If you specify to store the output information in a file, the following may appear.

- `Overwrite the output file? ([true], false)`

  Specify the way to handle the specified output file. Selecting **`true`** will overwrite the file. Selecting **`false`** will append the output information to the file. The default is **`true`**.

You can also execute the task in silent mode by executing one of the following commands.

— `ant -f uddiant.xml get_bindingDetail -DbindingKey=`*`BindingKey`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`

— uddiant `get_bindingDetail -DbindingKey=`*`BindingKey`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`

get_businessDetail

Execute one of the following commands in CLI to execute a get_businessDetail task in the interactive mode:

— `ant -f uddiant.xml get_businessDetail`

— uddiant `get_businessDetail`

The following prompts will appear.

- `Enter the Business Key (required)`

  Enter the UDDI key of the desired business entity.

- `Enter the output file location in the relative or absolute path`

  To store the output information in a file, enter the relative or absolute path and the name of the file.

  You can also display the output information in CLI by pressing **Enter**.

  If you specify to store the output information in a file, the following may appear.

- `Overwrite the output file? ([true], false)`

  Specify the way to handle the specified output file. Selecting **`true`** will overwrite the file. Selecting **`false`** will append the output information to the file. The default is **`true`**.

You can also execute the task in silent mode by executing one of the following commands.

— `ant -f uddiant.xml get_businessDetail -DbusinessKey=`*`BusinessKey`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`

— uddiant `get_businessDetail -DbusinessKey=`*`BusinessKey`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`

get_serviceDetail

Execute one of the following commands in CLI to execute a get_serviceDetail task in the interactive mode:

— `ant -f uddiant.xml get_serviceDetail`

— uddiant `get_serviceDetail`

The following prompts will appear.

- `Enter the Business Service Key (required)`

  Enter the UDDI key of the desired business service.

- `Enter the output file location in the relative or absolute path`

  To store the output information in a file, enter the relative or absolute path and the name of the file.

  You can also display the output information in CLI by pressing **Enter**.

  If you specify to store the output information in a file, the following may appear.

- `Overwrite the output file? ([true], false)`

  Specify the way to handle the specified output file. Selecting **`true`** will overwrite the file. Selecting **`false`** will append the output information to the file. The default is **`true`**.

You can also execute the task in silent mode by executing one of the following commands.

— `ant -f uddiant.xml get_serviceDetail -DserviceKey=`*`ServiceKey`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`

— uddiant `get_serviceDetail -DserviceKey=`*`ServiceKey`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`

get_tModelDetail

Execute one of the following commands in CLI to execute a get_tModelDetail task in the interactive mode:

— `ant -f uddiant.xml get_tModelDetail`

— uddiant `get_tModelDetail`

The following prompts will appear.

- `Enter the Tmodel Key(required)`

  Enter the UDDI key of the desired tModel.

- `Enter the output file location in the relative or absolute path`

  To store the output information in a file, enter the relative or absolute path and the name of the file.

  You can also display the output information in CLI by pressing **Enter**.

  If you specify to store the output information in a file, the following may appear.

- `Overwrite the output file? (true, [false])`

  Specify the way to handle the specified file. Selecting **`true`** will overwrite the specified file.

You can also execute the task in silent mode by executing one of the following commands.

— `ant -f uddiant.xml get_tModelDetail -DtmodelKey=`*tModelKey* `-Dfile=`*OutputFilePath/OutputFileName* `-Doverwrite=[true|false]`

— uddiant `get_tModelDetail -DtmodelKey=`*tModelKey* `-Dfile=`*OutputFilePath/OutputFileName* `-Doverwrite=[true|false]`

Save Tasks

Execute one of the following commands in CLI to execute a save task in the interactive mode:

— `ant -f uddiant.xml save_`*sampleActivity*

— uddiant `save_`*sampleActivity*

Each save task prompts whether to use a data file. The following prompts will appear in CLI.

- `Enter the data from the data file (y, n)?`

  Press **Y** and then press **Enter** to use a data file. Otherwise, press **N**.

  If you press **Y** and then press **Enter**, the following prompts will appear.

- `Enter the data file (required).`

  Enter the absolute path and the name of the data file.

- `Enter the output file location in the relative or absolute path`

  To store the output information of the task in a file, enter the absolute path or the relative path and the name of the file.

  You can also display the output information in CLI by pressing **Enter**.

  If you specify to store the output information in a file, the following may appear.

- `Overwrite the output file? ([true], false)`

  Specify the way to handle the specified output file. Selecting **`true`** will overwrite the file. Selecting **`false`** will append the output information to the file. The default is **`true`**.

save_binding

Execute one of the following commands in CLI to execute a save_binding task in the interactive mode:

— `ant -f uddiant.xml save_binding`

— uddiant `save_binding`

If you press **N** and then press **Enter** in response to the `Enter the data from the data file(y, n)` prompt, the following prompts will appear.

- `Enter the Service Key(required)`

Enter the binding service key.

- `Enter the Binding Key(Optional)`

  Enter the binding key if needed.

- `Enter the Service end point(required)`

  Enter the service access point (endpoint).

- `Enter the end point type: (required)`

  Enter the type of the service access point.

- `Enter the service WSDL URL:`

  Enter the Web Service Definition Language URL for the service.

- `Enter the output file location in the relative or absolute path`

  To store the output information of the task in a file, enter the absolute path or the relative path and the name of the file.

  You can also display the output information in CLI by pressing **Enter**.

  If you specify to store the output information in a file, the following may appear.

- `Overwrite the output file? ([true], false)`

  Specify the way to handle the specified output file. Selecting **`true`** will overwrite the file. Selecting **`false`** will append the output information to the file. The default is **`true`**.

You can also execute the task in silent mode by executing one of the following commands.

- To use the settings in a data file specified in the command line:

  — `ant -f uddiant.xml save_binding -Ddatafromfile=y -DdataFile=`*`DataFilePath/DataFileName`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`

  — uddiant `save_binding -Ddatafromfile=y -DdataFile=`*`DataFilePath\DataFileName`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`

- To use the settings specified in the command line:

  — `ant -f uddiant.xml save_binding -Ddatafromfile=n -DserviceKey=`*`ServiceKey`* `-DbindingKey=`*`BindingKey`* `-DaccessPoint=`*`AccessPoint`* `-DaccessPointUseType=`*`Endpoint`*

`-DwsdlURL=`*WSDL_URL* `-Dfile=`*OutputFilePath/OutputFileName* `-Doverwrite=[true|false]`

— uddiant `save_binding -Ddatafromfile=n -DserviceKey=`*ServiceKey* `-DbindingKey=`*BindingKey* `-DaccessPoint=`*AccessPoint* `-DaccessPointUseType=`*Endpoint* `-DwsdlURL=`*WSDL_URL* `-Dfile=`*OutputFilePath/OutputFileName* `-Doverwrite=[true|false]`

For information about the structure of the data files used by save_binding tasks, see Section 5.2.15 save_binding in *UDDI Version 3.0.2* at http://www.uddi.org/pubs/uddi_v3.htm#_Toc85908110.

A sample data file `publish_binding_data.xml` is provided in the `TIBCO_HOME/RuntimeUDDIServer/3.1/sample/data` directory.

save_business

Execute one of the following commands in CLI to execute a save_business task in the interactive mode:

— `ant -f uddiant.xml save_business`

— uddiant `save_business`

If you press **N** and then press **Enter** in response to the `Enter the data from the data file (y, n)` prompt, the following prompts will appear.

- `Enter the Business Name(required)`

  Enter the business name.

- `Enter the description:(Optional)`

  Enter a description for the Business.

- `Enter the Business Key(Optional):`

  Enter the business key if needed.

- `Enter the output file location in the relative or absolute path`

  To store the output information of the task in a file, enter the absolute path or the relative path and the name of the file.

  You can also display the output information in CLI by pressing **Enter**.

  If you specify to store the output information in a file, the following may appear.

- `Overwrite the output file? ([true], false)`

Specify the way to handle the specified output file. Selecting **`true`** will have the file overwritten. Selecting **`false`** will append the output information to the file. The default is **`true`**.

You can also have the task executed in the silent mode by executing one of the following commands.

- To use the settings in a data file specified in the command line:
  - `ant -f uddiant.xml save_business -Ddatafromfile=y -DdataFile=`*`DataFilePath/DataFileName`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`
  - uddiant `save_business -Ddatafromfile=y -DdataFile=`*`DataFilePath/DataFileName`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`
- To use the settings specified in the command line:
  - `ant -f uddiant.xml save_business -Ddatafromfile=n -DbusinessName=`*`BusinessName`* `-Ddescription=`*`Description`* `-DbusinessKey=`*`BusinessKey`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`
  - `uddiant save_business -Ddatafromfile=n -DbusinessName=`*`BusinessName`* `-Ddescription=`*`Description`* `-DbusinessKey=`*`BusinessKey`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`

For information about the structure of the data files used by save_business tasks, see Section 5.2.16 save_business in *UDDI Version 3.0.2* at http://www.uddi.org/pubs/uddi_v3.htm#_Toc85908111.

A sample data file `publish_business_data.xml` is provided in the `TIBCO_HOME/RuntimeUDDIServer/3.1/sample/data` directory.

save_service

Execute one of the following commands in CLI to execute a save_service task in the interactive mode:

- `ant -f uddiant.xml save_service`
- uddiant `save_service`

If you press **N** and then press **Enter** in response to the `Enter the data from the data file (y, n)` prompt, the following prompts will appear.

- `Enter the Service Name (required)`

  Enter the service name.

- `Enter the description:(Optional)`

  Enter a description for the service.

- `Enter the Business Key(required)`

  Enter the business key for the service.

- `Enter the Service Key(optional):`

  Enter the service key if needed.

- `Enter the service Endpoint: (required)`

  Enter the service access point (endpoint).

- `Enter the end point type:`

  Enter the type of the endpoint. See *UDDI Spec V3.0* for the possible values.

- `Enter the service WSDL URL:`

  Enter the Web Service Definition Language URL for the service.

- `Enter the output file location in the relative or absolute path`

  To store the output information of the task in a file, enter the absolute path or the relative path and the name of the file.

  You can also display the output information in CLI by pressing **Enter**.

  If you specify to store the output information in a file, the following may appear.

- `Overwrite the output file? ([true], false)`

  Specify the way to handle the specified output file. Selecting **`true`** will overwrite the file. Selecting **`false`** will append the output information to the file. The default is **`true`**.

You can also execute the task in silent mode by executing one of the following commands.

- To use the settings in a data file specified in the command line:

  — `ant -f uddiant.xml save_service -Ddatafromfile=y -DdataFile=`*`DataFilePath/DataFileName`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`

  — uddiant `save_service -Ddatafromfile=y -DdataFile=`*`DataFilePath/DataFileName`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`

- To use the settings specified in the command line:

  — `ant -f uddiant.xml save_service -Ddatafromfile=n -DserviceName=`*ServiceName* `-Ddescription=`*Description* `-DbusinessKey=`*BusinessKey* `-DaccessPointUseType=`*Endpoint* `-Dendpoint=`*EndPoint_URL* `-DwsdlURL=`*WSDL_URL* `-DserviceKey=`*ServiceKey* `-Dfile=`*OutputFilePath/OutputFileName* `-Doverwrite=[true|false]`

  — uddiant `save_service -Ddatafromfile=n -DserviceName=`*ServiceName* `-Ddescription=`*Description* `-DbusinessKey=`*BusinessKey* `-DaccessPointUseType=`*Endpoint* `-Dendpoint=`*EndPoint_URL* `-DwsdlURL=`*WSDL_URL* `-DserviceKey=`*ServiceKey* `-Dfile=`*OutputFilePath/OutputFileName* `-Doverwrite=[true|false]`

For information about the structure of the data files used by save_service tasks, see Section 5.2.17 save_service in *UDDI Version 3.0.2* at http://www.uddi.org/pubs/uddi_v3.htm#_Toc85908112.

A sample data file `publish_service_data.xml` is provided in the `TIBCO_HOME/RuntimeUDDIServer/3.1/sample/data` directory.

save_tModel

Execute one of the following commands in CLI to execute a save_tModel task in the interactive mode:

— `ant -f uddiant.xml save_tModel`

— uddiant `save_tModel`

If you press **N** and then press **Enter** in response to the `Enter the data from the data file (y, n)` prompt, the following prompts will appear.

- `Enter the Tmodel Name(required)`

  Enter a name for the tModel.

- `Enter the Tmodel Key(optional):`

  Enter the tModel key if needed.

- `Enter the output file location in the relative or absolute path`

  To store the output information of the task in a file, enter the absolute path or the relative path and the name of the file.

  You can also display the output information in CLI by pressing **Enter**.

  If you specify to store the output information in a file, the following may appear.

- `Overwrite the output file? ([true], false)`

  Specify the way to handle the specified output file. Selecting **`true`** will overwrite the file. Selecting **`false`** will append the output information to the file. The default is **`true`**.

You can also execute the task in silent mode by executing one of the following commands:

- To use the settings in the data file specified in the command line:
  - `ant -f uddiant.xml save_tModel -Ddatafromfile=y -DdataFile=`*`DataFilePath/DataFileName`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`
  - uddiant `save_tModel -Ddatafromfile=y -DdataFile=`*`DataFilePath/DataFileName`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`
- To use the settings specified in the command line:
  - `ant -f uddiant.xml save_tModel -Ddatafromfile=n -DtmodelName=`*`tModelName`* `-DtmodelKey=`*`tModedlKey`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`
  - uddiant `save_tModel -Ddatafromfile=n -DtmodelName=`*`tModelName`* `-DtmodelKey=`*`tModedlKey`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`

For information about the structure of the data files used by save_tModel tasks, see Section 5.2.18 save_tModel in *UDDI Version 3.0.2* at http://www.uddi.org/pubs/uddi_v3.htm#_Toc85908113.

A sample data file `publish_tModel_data.xml` is provided in the `TIBCO_HOME/RuntimeUDDIServer/3.1/sample/data` directory.

Delete Tasks

Each delete task prompts for the key of the entity to be deleted and does not return any result.

delete_binding

Execute one of the following commands in CLI to execute a delete_binding task in the interactive mode:

— `ant -f uddiant.xml delete_binding`

— uddiant `delete_binding`

The following prompt will appear.

- `Enter the Binding Key(required)`

  Enter the binding key of the binding template to be deleted.

You can also execute the task in silent mode by executing one of the following commands:

— `ant -f uddiant.xml delete_binding -DbindingKey=`*BindingKey*

— uddiant `delete_binding -DbindingKey=`*BindingKey*

delete_business

Execute one of the following commands in CLI to execute a delete_business task in the interactive mode:

— `ant -f uddiant.xml delete_business`

— uddiant `delete_business`

The following prompt will appear.

- `Enter the Business Key(required)`

  Enter the business key of the business to be deleted.

You can also execute the task in silent mode by executing one of the following commands:

— `ant -f uddiant.xml delete_business -DbusinessKey=`*BusinessKey*

— uddiant `delete_business -DbusinessKey=`*BusinessKey*

delete_service

Execute one of the following commands in CLI to execute a delete_service task in the interactive mode:

— `ant -f uddiant.xml delete_service`

— uddiant `delete_service`

The following prompt will appear.

- `Enter the Service Key(required)`

  Enter the service key of the service to be deleted.

You can also execute the task in silent mode by executing one of the following commands:

— `ant -f uddiant.xml delete_service -DserviceKey=`*ServiceKey*

— uddiant `delete_service -DserviceKey=`*ServiceKey*

delete_tModel

Execute one of the following commands in CLI to execute a delete_tModel task in the interactive mode:

— `ant -f uddiant.xml delete_tModel`

— uddiant `delete_tModel`

The following prompt will appear.

- `Enter the Tmodel Key(required)`

  Enter the tModel key of the tModel to be deleted.

You can also execute the task in silent mode by executing one of the following commands:

— `ant -f uddiant.xml delete_tModel -DtmodelKey=`*tModelKey*

— uddiant `delete_tModel -DtmodelKey=`*tModelKey*

Subscription Tasks

You can perform the following Subscription tasks through CLI. For detailed information, see the related sections in *TIBCO ActiveMatrix Runtime UDDI Server Administrator's Guide*.

get_subscriptions

Execute one of the following commands in CLI to execute a get_subscriptions task in the interactive mode:

— `ant -f uddiant.xml get_subscriptions`

— uddiant `get_subscriptions`

The following prompts will appear.

- `Enter the output file location in the relative or absolute path`

  To store the output information of the task in a file, enter the absolute path or the relative path and the name of the file.

  You can also display the output information in CLI by pressing **Enter**.

  If you specify to store the output information in a file, the following may appear.

- `Overwrite the output file? ([true], false)`

  Specify the way to handle the specified output file. Selecting **`true`** will overwrite the file. Selecting **`false`** will append the output information to the file. The default is **`true`**.

You can also execute the task in silent mode by executing one of the following commands:

— `ant -f uddiant.xml get_subscriptions -Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`

— uddiant `get_subscriptions -Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`

get_subscriptionResults

Execute one of the following commands in CLI to execute a get_subscriptionResults task in the interactive mode:

— `ant -f uddiant.xml get_subscriptionResults`

— uddiant `get_subscriptionResults`

The following prompt will appear.

`Enter the data from the data file (y, n)?`

If you press **Y** and then press **Enter**, the following prompts will appear.

- `Enter the data file..(required)`

  Enter the full path and the name of the data file.

- `Enter the output file location in the relative or absolute path`

  To store the output information of the task in a file, enter the absolute path or the relative path and the name of the file.

  You can also display the output information in CLI by pressing **Enter**.

  If you specify to store the output information in a file, the following may appear.

- `Overwrite the output file? ([true], false)`

  Specify the way to handle the specified output file. Selecting **`true`** will overwrite the file. Selecting **`false`** will append the output information to the file. The default is **`true`**.

If you press **N** and then press **Enter** in response to the `Enter the data from the data file (y, n)` prompt, the following prompts will appear.

- `Enter the Subscription Key(required)`

  Enter the key of a specific Subscription to be retrieved.

- `Enter the StartPoint of CoveragePeriod (optional/format:2008-08-08T08:08:08.000Z)`

  Enter the start point of the retrieval period.

- `Enter the EndPoint of CoveragePeriod (optional/format:2008-08-08T08:08:08.000Z)`

  Enter the end point of the retrieval period.

- `Enter the ChunkToken (optional)`

  Enter the Chunk Token.

- `Enter the output file location in the relative or absolute path`

To store the output information of the task in a file, enter the absolute path or the relative path and the name of the file.

You can also display the output information in CLI by pressing **Enter**.

If you specify to store the output information in a file, the following may appear.

- `Overwrite the output file? (true, [false])`

  Specify the way to handle the specified file. Selecting **`true`** will overwrite the data file. The default is **`false`**.

You can also execute the task in silent mode by executing one of the following command:

— `ant -f uddiant.xml get_subscriptionResults -Ddatafromfile=n -DsubscriptionKey=`*`SubscriptionKey`* `-DstartPoint=`*`StartPoint`* `-DendPoint=`*`EndPoint`* `-DchunkToken=`*`ChunkToken`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`

— uddiant `get_subscriptionResults -Ddatafromfile=n -DsubscriptionKey=`*`SubscriptionKey`* `-DstartPoint=`*`StartPoint`* `-DendPoint=`*`EndPoint`* `-DchunkToken=`*`ChunkToken`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`

For information about the structure of the data files used by get_subscriptionResults tasks, see Section 5.5.11 get_subscriptionResults in *UDDI Version 3.0.2* at http://www.uddi.org/pubs/uddi_v3.htm#_Toc85908139.

Currently, the get_subscriptionResults task cannot retrieve information about deleted entities. You can use the Polling Subscription API or adopt the asynchronous notification mechanism to get information about deleted entities. See Polling Subscription API on page 81 for more information.

save_subscription

Execute one of the following commands in CLI to execute a save_subscription task in the interactive mode:

— `ant -f uddiant.xml save_subscription`

— uddiant `save_subscription`

If you press **N** and then press **Enter** in response to the `Enter the data from the data file (y, n)` prompt, the following prompts will appear.

- `Please choose one get_xx or find_xx API to input the subscription data:`

  ```
  (1): find_binding
  (2): find_business
  (3): find_service
  (4): find_tModel
  (5): get_bindingDetail
  (6): get_businessDetail
  (7): get_serviceDetail
  (8): get_tModelDetail
  ```

  `Your Option: ([1], 2, 3, 4, 5, 6, 7, 8)`

  Select the API to be used for the subscription task as prompted. The default is **`1`**.

- `Enter the Business Service Key(optional)`

  Enter the business service key.

- `Enter the Subscription Key (optional)`

  Enter the subscription key if needed.

- `Enter the Subscription Binding Key (optional)`

  Enter the subscription binding key if needed.

- `Enter the Brief (optional) (true, [false])`

  Select **`true`** to enter the brief. The default is **`false`**.

- `Enter theEnter the NotificationInterval(optional)`

  Enter the notification interval. For related information, see Section 5.5.2 Specifying Durations in *UDDI Version 3.0.2* at http://www.uddi.org/pubs/uddi_v3.htm#_Toc85908130.

- `Enter the MaxEntities(optional)`

  Enter the maximum entity.

- `Enter the ExpiresAfter (optional/format:2008-08-08T08:08:08.000Z)`

  Enter the date and time (UTC time) when the subscription will expire.

- `Enter the output file location in the relative or absolute path`

  To store the output information of the task in a file, enter the absolute path or the relative path and the name of the file.

  You can also have the output information displayed in CLI by pressing **Enter**.

If you specify to store the output information in a file, the following may appear.

- `Overwrite the output file? ([true], false)`

  Specify the way to handle the specified output file. Selecting **`true`** will overwrite the file. Selecting **`false`** will append the output information to the file. The default is **`true`**.

You can also execute the task in silent mode by executing one of the following commands:

- To use the settings in the data file specified in the command line:

  — `ant -f uddiant.xml save_subscription -Ddatafromfile=y -DdataFile=`*`DataFilePath\DataFileName`* `-Dfile=`*`OutputFilePath\OutputFileName`* `-Doverwrite=[true|false]`

  — uddiant `save_subscription -Ddatafromfile=y -DdataFile=`*`DataFilePath/DataFileName`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`

- To use the settings specified in the command line:

  — `ant -f uddiant.xml save_subscription -Ddatafromfile=n -Dsubscription.filter.type=[1|2|3|4|5|6|7|8] -DserviceKey=`*`ServiceKey`* `-DsubscriptionKey=`*`SubscriptionKey`* `-DsubBindingKey=`*`SubscriptinBindingKey`* `-Dbrief=[true|false] -DnotificationInterval=`*`NotificationInterval`* `-DmaxEntities=`*`MaxEntities`* `-DexpireAfter=`*`ExpirationTime`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`

  — uddiant `save_subscription -Ddatafromfile=n -Dsubscription.filter.type=[1|2|3|4|5|6|7|8] -DserviceKey=`*`ServiceKey`* `-DsubscriptionKey=`*`SubscriptionKey`* `-DsubBindingKey=`*`SubscriptinBindingKey`* `-Dbrief=[true|false] -DnotificationInterval=`*`NotificationInterval`* `-DmaxEntities=`*`MaxEntities`* `-DexpireAfter=`*`ExpirationTime`* `-Dfile=`*`OutputFilePath/OutputFileName`* `-Doverwrite=[true|false]`

  The `subscription.filter.type` property specifies the API to be used, as described at the beginning of save_subscription on page 74.

For information about the structure of the data files used by save_subscription tasks, see Section 5.5.8 save_subscription in *UDDI Version 3.0.2* at http://www.uddi.org/pubs/uddi_v3.htm#_Toc85908136.

delete_subscription

Execute one of the following commands in CLI to execute a delete_subscription task in the interactive mode:

- `ant -f uddiant.xml delete_subscription`
- uddiant `delete_subscription`

The following prompt will appear.

- `Enter the Subscription Key(required)`

  Enter the subscription key to be deleted.

You can also execute the task in silent mode by executing one the following commands:

- `ant -f uddiant.xml delete_subscription -DsubscriptionKey=`*SubscriptionKey*
- uddiant `delete_subscription -DsubscriptionKey=`*SubscriptionKey*

Security Tasks

The security tasks include the connect and disconnect tasks. The tasks are used when invoking multiple tasks in a single target. They only provide registry URLs and credentials required to connect to a UDDI server. After the connection to the server is established, all the other tasks use the same credentials till the connection is terminated.

The connect task takes the `connectionid` property, which is passed to all the other tasks as the connection information. While executing multiple tasks in a target, you can provide the `resultKeys` property to all find tasks, which will save all UDDI keys in the property. As shown in the following example, the property is passed to all get tasks to retrieve the particular entity.

```
<find_business
      businessName="node"
      qualifier="exactMatch"
      outputFile="OutputFilePath\OutputFileName"
      limit="5"
      overwrite="false"
      resultKeys="businessKey"
      connectionid="ConnectionID"/>
```

The output of the above task will be stored in the output file and the UDDI keys of all the business entities returned are stored in the property specified in the `resultKeys` property (in this example, businessKey is specified). You can use `resultKeys` as a parameter to get the business entities through the following `get_businessDetail` task.

```
<get_businessDetail
      businessKey="${businessKey}"
      outputFile="OutputFilePath\OutputFileName"
      connectionid=ConnectionID/>
```

The following example is used to connect to a server. It contains all the related properties. In actual use, you can set part of the properties as needed.

```
<connect userName=UserName password=Password
      inquiryURL="http://Host:Port/uddi/services/inquiry"
      securityURL="http://Host:Port/uddi/services/security"
      publishURL="http://Host:Port/uddi/services/publication"
      subscriptionURL="http://Host:Port/uddi/services/subscription"
      adminURL="http://Host:Port/uddi/services/admin"
      trustStoreFolder=CertificatePath
      trustStorePath=TrustStorePath
      trustStoreType=TrustStoreType
      trustStorePassword=TrustStorePassword
      proxyHost=ProxyHostAddress
      proxyPort=ProxyPort
      proxyUsername=ProxyUsername
      proxyPassword=ProxyPassword
```

```
    connectionid=ConnectionID/>
```

With anonymous inquiry disabled, the `username`, `password`, and `securityURL` properties are required; with anonymous inquiry enabled, the `username` and `password` properties can be left empty. See Table 3, Ant-based Tasks and the Corresponding Required URL for related information.

You can execute multiple tasks in a target, as shown below.

```
<taskdef resource="com/tibco/uddi/commandline/ant/antlib.xml" />
<target name="executeMultiple">

  <connect
      userName=Username
      password=Password
      inquiryURL="http://Host:Port/uddi/services/inquiry"
      securityURL="http://Host:Port/uddi/services/security"
      publishURL="http://Host:Port/uddi/services/publication"
      subscriptionURL="http://Host:Port/uddi/services/subscription"
      connectionid=ConnectionID/>

  <find_business
      businessName="node"
      qualifier="exactMatch"
      outputFile="OutputFilePath\OutputFileName"
      limit="5"
      overwrite="false"
      resultKeys="businessKey"
      connectionid=ConnectionID/>

  <get_businessDetail
      businessKey="${businessKey}"
      outputFile="OutputFilePath\OutputfileName"
      connectionid=ConnectionID/>

  <find_tModel
      tmodelName="%"
      qualifier="approximateMatch"
      outputFile="OutputFilePath\OutputfileName"
      overwrite="false"
      failOnError="true"
      resultKeys="tmodelKey"
      limit="5"
      connectionid=ConnectionID/>

  <get_tModelDetail
      tmodelKey="${tmodelKey}"
      overwrite="false"
      outputFile="OutputFilePath\OutputFileName"
      connectionid=ConnectionID/>

  <save_business
      businessName="Name"
      description="description"
```

```
        businessKey=BusinessKey
        outputFile="OutputFilePath\OutputFileName"
        overwrite="false"
        connectionid=ConnectionID/>

    <delete_business
        businessKey=BusinessKey
        connectionid=ConnectionID/>

    <save_subscription
        dataType="find_business"
        businessName="%"
        qualifier="approximateMatch"
        subscriptionKey=SubscriptionKey
        subBindingKey=""
        brief="false"
        notificationInterval=NotificationInterval
        maxEntities="5"
        expireAfter=ExpireTime
        outputFile="OutputFilePath\OutputFileName"
        overwrite="false"
     connectionid=ConnectionID/>
     <disconnect connectionid=ConnectionID/>
</target>
```

- There are two ways to specify the API used to input subscription data for the save_subscription Ant-based task. To specify the API in data files, you need to specify the name of the API (such as `find_business` in the above example). To specify the API in CLI, however, you need to specify the number corresponding to the desired API, as shown in the beginning of save_subscription on page 74.
- When executing multiple tasks in a target, you need to make sure the output files of different Ant-based tasks are different to prevent output data of different Ant-based tasks from being overwritten. Otherwise, even if the overwrite option is set to false, the output of a subsequent Ant-based task will overwrite that of the previous one. For example, the output file for the find_business and get_businessDetail tasks in the above sample should be different.

Chapter 4 Polling Subscription API

This chapter describes how to use the Polling Subscription API.

Topics

Introduction

The common way to monitor changes in a UDDI registry is to create a subscription and save it in the UDDI registry. When creating a subscription, you can specify the entities you want to monitor by setting searching criteria for the subscription, through which only the entities matching the searching criteria are monitored. You can also set the time period when the subscription will be valid. Once a subscription is created and saved, the UDDI server will record the changes of the entities matching the search criteria in the specified time period and will stop recording when the subscription period expires.

After you save a subscription in the registry, the UDDI server normally adopts one of the two mechanisms to notify you of entity changes: the synchronous tracking mechanism and the asynchronous notification mechanism. When adopting the synchronous tracking mechanism, a client obtains information about entity changes by sending HTTP-based SOAP messages to the UDDI server. When adopting the asynchronous notification mechanism, however, you need to create an HTTP-based web service and register the web service endpoint in the registry as a notification listener, or register an e-mail address in the registry. The UDDI registry will then periodically send information about entity changes to the listener service or to the registered e-mail address.

TIBCO ActiveMatrix Runtime UDDI Server supports both mechanisms. The `get_subscriptionResults` API is the standard API for tracking entity changes synchronously. As the API does not have a wrapper level and it requires the start time and end time each time it is called, TIBCO provides a proprietary API based on this API to manage subscriptions and the start/end time. The API is known as Polling Subscription API. It provides an easy way to call the `get_subscriptionResults` API.

For information about the standard subscription APIs, see Section 5.5 Subscription API Set and Appendix C Supporting Subscribers in *UDDI Version V3.0.2* at http://www.uddi.org/pubs/uddi_v3.htm#_Toc85908128 and http://www.uddi.org/pubs/uddi_v3.htm#_Toc85908401.

Requirements for Using Polling Subscription API

You need to provide your own `NotificationHandler` implementation. The entity changes are in `subscriptionResultList` and you need to decide what you want to do with the changes. The subscription result list contains the snapshot of the entities in the registry the last time the `sendResult` method was invoked. You need to compare the result list the last time the `sendResult` method was invoked with the current one to see if changes are made to the entities.

You need also to create an instance of the `PollingSubscription` class and set its parameters through the `ConfigurationInfo` class. The `NotificationHandler` interface, `PollingSubscription`, and `ConfigurationInfo` classes are all in the package named `com.tibco.uddi.subscription`. For information on the use of these classes, see the corresponding documents in the JAVADoc directory.

To avoid an auth token expiration exception, you should set the polling interval shorter than the token expiration time.

For information on other APIs shipped with TIBCO ActiveMatrix Runtime UDDI Server, see the related JAVA documentation in the JAVADoc directory: `file:///`*`UDDI_HOME`*`/doc/api/index.html`

Developing and Implementing Your Own Subscription Notification Handler

This section describes how to develop and implement your own subscription notification handler.

Importing Polling Subscription API

You can import Polling Subscription API in one of the following two ways.

The build numbers of the following plug-ins vary with product versions.

Developing a Your Own Subscription Notification Handler

The subscription notification handler can be developed as an Eclipse plug-in or a JAVA program.

- **Developing the subscription notification handler as an Eclipse plug-in**

  Table 4 lists the required packages and the corresponding plug-ins. All the plug-ins mentioned are located in *TIBCO_HOME*\components\shared\1.0.0\plugins.

Table 4 Packages and the Corresponding Plug-ins

Package required	Corresponding plug-in
com.tibco.registry.uddi.subscription;version="[3.0.0, 4.0.0)"	com.tibco.registry.uddi.subscription_3.1.0.003jar
dk.itst.uddi.client;version="[1.3.0,2.0.0)" dk.itst.uddi.client.exceptions;version="[1.3.0,2.0.0)" dk.itst.uddi.client.query;version="[1.3.0,2.0.0)" dk.itst.uddi.client.stubs;version="[1.3.0,2.0.0)" dk.itst.uddi.client.types.core;version="[1.3.0,2.0.0)" dk.itst.uddi.client.types.sub;version="[1.3.0,2.0.0)"	com.tibco.tpcl.dk.itst.uddi.client.1.4.0.001
javax.wsdl;version="[1.1.100,2.0.0)"	com.tibco.tpcl.javax.wsdl4j_1.6.100.001

Table 4 Packages and the Corresponding Plug-ins

Package required	Corresponding plug-in
org.apache.axiom.om.impl;version="[2.0.100,3.0.0)"	com.tibco.tpcl.org.apache.axiom_2.0.100.001-1_2_7
org.apache.axis2.client;version="[2.1.100,3.0.0)"	com.tibco.tpcl.org.apache.axis2.runtime_1.4.100.001
org.apache.commons.codec;version="[1.3.100,2.0.0)"	com.tibco.tpcl.org.apache.commons.codec_1.3.100.001
org.apache.commons.httpclient;version="[3.1.100,4.0.0)"	com.tibco.tpcl.org.apache.commons.httpclient_3.1.100.001
org.apache.log4j;version="[1.2.200,2.0.0)"	com.tibco.tpcl.org.apache.log4j_1.2.200.003
org.apache.neethi;version="[2.0.4,3.0.0)"	com.tibco.tpcl.org.apache.neethi_2.0.4.001
org.apache.ws.commons.schema.resolver;version="[2.0.100,3.0.0)"	com.tibco.tpcl.org.apache.ws.commons.schema_2.0.100.001
org.apache.xmlbeans;version="[2.4.100,3.0.0)"	com.tibco.tpcl.org.apache.xmlbeans_2.4.100.003
org.slf4j.impl;version="[1.5.2,2.0.0)"	com.tibco.tpcl.org.apache.commons.logging.over.slf4j_1.5.100.002
javax.activation; version="[1.1.100,2.0.0)"	com.tibco.tpcl.javax.activation_1.1.100.001
javax.xml.stream;version="[1.0.0,2.0.0)"	com.tibco.tpcl.javax.osgi.factories_1.0.0.003.jar
javax.xml.namespace;version="[1.1.0,2.0.0)"	com.tibco.tpcl.javax.system.exports_5.0.100.003

- **Developing the subscription notification handler as a JAVA program**

  You need to add the paths of the following JAR files to the classpath. You can find the these JAR files in the *TIBCO_HOME*\components\shared\1.0.0\plugins directory.

  com.tibco.registry.uddi.subscription_3.1.0.003.jar

  com.tibco.tpcl.dk.itst.uddi.client_1.4.0.001/openuddi-client-1.4-patched.jar

  com.tibco.tpcl.javax.system.exports_5.0.100.003/xml-apis.jar

  com.tibco.tpcl.javax.osgi.factories_1.0.0.003.jar

com.tibco.tpcl.javax.wsdl4j_1.6.100.001/wsdl4j.jar

com.tibco.tpcl.org.apache.axiom_2.0.100.001-1_2_7/axiom-impl-1.2.7.jar

com.tibco.tpcl.org.apache.axis2.runtime_1.4.100.001/axis2-kernel-1.4.1.jar

com.tibco.tpcl.org.apache.commons.codec_1.3.100.001/commons-codec-1.3.jar

com.tibco.tpcl.org.apache.commons.httpclient_3.1.100.003/commons-httpclient-3.1.jar

com.tibco.tpcl.org.apache.log4j_1.2.200.003/log4j-1.2.15.jar

com.tibco.tpcl.org.apache.neethi_2.0.4.001/neethi-2.0.4.jar

com.tibco.tpcl.org.apache.ws.commons.schema_2.0.100.001/XmlSchema-1.4.2.jar

com.tibco.tpcl.org.apache.xmlbeans_2.4.100.003/xbean.jar

com.tibco.tpcl.org.apache.commons.logging.over.slf4j_1.5.2.003/jcl-over-slf4j-1.5.2.jar

com.tibco.tpcl.javax.activation_1.1.100.001/activation.jar

JAVADoc for Polling Subscription API

See the files in *TIBCO_HOME*/RuntimeUDDIServer/3.1/doc/api/.

Sample

This section provides a sample that shows how to use Polling Subscription API.

Obtaining the Sample File

- The sample source code is given in the `DefaultNotificationHandler.java` file and the `SamplePollingSubscriptionDriver.java` file in *TIBCO_HOME*`/RuntimeUDDIServer/3.1/sample/api/src/sample/`.
- The executable file, `pollingsub.exe`, is in *TIBCO_HOME*`/RuntimeUDDIServer/3.1/sample/api/bin/`.
- The `.project` file for Eclipse is in *TIBCO_HOME*`/RuntimeUDDIServer/3.1/sample/api/`.

Running the Sample

To run the sample:

1. Open a command line window and change the current path to *TIBCO_HOME*`/RuntimeUDDIServer/3.1/sample/api/bin`.
2. Run `pollingsub.exe` to execute the sample `SamplePollingSubscriptionDriver.java`.

- Update the `pollingsub.properties` file to make it consistent with your UDDI server configuration.
- Check the `pollingsub.log` log file to get the running information of the sample.

  By default, the log file is located in the following directory: *configDirectoryRoot*`/tibco/cfgmamt/RuntimeUDDIServer/logs`.

  To change the location of the log file, modify the corresponding content in the following file:

  TIBCO_HOME`/RuntimeUDDIServer/3.1/cfg/pullingsub-log4j.properties`
- The file containing the subscription results is in the directory indicated by the `JRE system.io.tmpdir` property.
  In Windows XP, the directory is *System_Install_Disk*`/Documents and Settings/`*LoginUser*`/Local Settings/Temp/`. The file is *subscriptionKey*`*.xml`.

3. To run your own classes:
   a. Rename a copy of the `pollingsub.tra` file (for example, rename the file as `sample.tra)`, replace the `SamplePollingSubscriptionDriver` class with the new class that contains the `main` method, and make sure the file containing your class is in the `tibco.flattener.class.path` property.
   b. Renamed a copy of the `pollingsub.exe` file to the name of the `.tra` file. For example, if `sample.tra` is in the directory, then rename the file as `sample.exe`.
   c. Execute the sample EXE file.

Chapter 5 Using the HTTP GET Service

This chapter describes how to use the HTTP GET service.

Topics

Introduction

The HTTP GET service provided by a UDDI node enables you to retrieve XML and HTML representations of UDDI data structures.

TIBCO ActiveMatrix Runtime UDDI Server enhances the HTTP GET service. It can transform the original response from XML format to XHTML format. With HTTP GET services enabled on a UDDI node, UDDI keys (such as businessKey, serviceKey, bindingKey, and tModelKey) in the XML responses will be transformed to HTML-based URLs. You can insert the URLs as hyper-links in web pages, emails, documents, and so on. You can then access the corresponding UDDI entity information simply by clicking the hyper-links. For more information, see Section 6.5 HTTP GET Services for UDDI Data Structures in *UDDI Version 3.0.2* at http://www.uddi.org/pubs/uddi_v3.htm#_Toc85908158.

HTTP GET services involve tasks requiring secured inquiry_URLs. So, if anonymous inquiry is disabled and you invoke an HTTP GET service, authentication is performed. In this case, you will be prompted to enter the username and the password.

Getting XHTML Responses

The following sections describe how to get UDDI entity information in XHTML format. To get a response in XHTML format, use a static URL in the format of `http://Host:Port/uddi/EntityType/EntityKey/html` or `http://Host:Port/uddi/EntityType/EntityKey`.

Getting Information about a Business Entity in XHTML Format

To get information about a business entity in XHTML format, you need to obtain the UDDI key of the entity and then insert the UDDI key into the above-mentioned URLs. For example, to get the information about the business entity with a UDDI key of `uddi:mydomain.com:39bb34ad-54e6-40e0-be2a-bd544e6772a4`, you need to:

- Replace *Host* and *Port* with the host and port of the TIBCO ActiveMatrix Runtime UDDI Server, for example, **`localhost:58080`**.
- Replace *EntityType* with **`business`**;
- Replace *EntityKey* with **`uddi:mydomain.com:39bb34ad-54e6-40e0-be2a-bd544e6772a4`**.

The following example URLs can be used to get the information about the business entity.

- `http://localhost:58080/uddi/business/uddi:mydomain.com:39bb34ad-54e6-40e0-be2a-bd544e6772a4/html`
- `http://localhost:58080/uddi/business/uddi:mydomain.com:39bb34ad-54e6-40e0-be2a-bd544e6772a4.`

Getting Information about a Service Entity in XHTML Format

To get information about a service entity in XHTML format, you need to obtain the UDDI key of the entity and then insert the UDDI key into the above-mentioned URLs. For example, to get the information about the service entity with a UDDI key of `uddi:mydomain.com:0e050e6d-cb3d-480e-8f13-a08fec5d7861`, you need to:

- Replace *Host* and *Port* with the host and port of the TIBCO ActiveMatrix Runtime UDDI Server, for example, **`localhost:58080`**.
- Replace *EntityType* with **`service`**;
- Replace *EntityKey* with **`uddi:mydomain.com:0e050e6d-cb3d-480e-8f13-a08fec5d7861`**.

The following example URLs can be used to get the information about the service entity.

- `http://localhost:58080/uddi/service/uddi:mydomain.com:0e050e6d-cb3d-480e-8f13-a08fec5d7861/html`
- `http://localhost:58080/uddi/service/uddi:mydomain.com:0e050e6d-cb3d-480e-8f13-a08fec5d7861.`

Getting Information about a Binding Template in XHTML Format

To get information about a binding template entity in XHTML format, you need to obtain the UDDI key of the entity and then insert the UDDI key into the above-mentioned URLs. For example, to get the information about the binding template entity with a UDDI key of `uddi:mydomain.com:d2e8705f-3449-4fa7-888c-c9b85ef127d8`, you need to:

- Replace *Host* and *Port* with the host and port of the TIBCO ActiveMatrix Runtime UDDI Server, for example, **`localhost:58080`**.
- Replace *EntityType* with **`binding`**;
- Replace *EntityKey* with **`uddi:mydomain.com:d2e8705f-3449-4fa7-888c-c9b85ef127d8`**.

The following example URLs can be used to get the information about the binding template entity.

- `http://localhost:58080/uddi/binding/uddi:mydomain.com:d2e8705f-3449-4fa7-888c-c9b85ef127d8/html`
- `http://localhost:58080/uddi/binding/uddi:mydomain.com:d2e8705f-3449-4fa7-888c-c9b85ef127d8.`

Getting Information about a tModel in XHTML Format

To get information about a tModel entity in XHTML format, you need to obtain the UDDI key of the entity and then insert the UDDI key into the above-mentioned URLs. For example, to get the information about the tModel entity with a UDDI key of `uddi:mydomain.com:bebc4299-3bc9-4975-9716-badfa9a2ce1c`, you need to:

- Replace *Host* and *Port* with the host and port of the TIBCO ActiveMatrix Runtime UDDI Server, for example, **`localhost:58080`**.
- Replace *EntityType* with **`tModel`**;
- Replace *EntityKey* with `uddi:mydomain.com:bebc4299-3bc9-4975-9716-badfa9a2ce1c`.

The following example URLs can be used to get the information about the binding template entity.

- `http://localhost:58080/uddi/tModel/uddi:mydomain.com:bebc4299-3bc9-4975-9716-badfa9a2ce1c/html`
- `http://localhost:58080/uddi/tModel/uddi:mydomain.com:bebc4299-3bc9-4975-9716-badfa9a2ce1c.`

Getting XML Responses

The following sections describe how to get UDDI entity information in XML format. To get a response in XML format, use a static URL in the format of `http://`*Host*`:`*Port*`/uddi/`*EntityType*`/`*EntityKey*`/xml` or use the HTTP GET request URL in the format of `http://`*Host*`:`*Port*`/uddi/http_get?`*EntityType*`Key=`*EntityKey*.

Getting Information about a Business Entity in XML Format

To get information about a business entity in XML format, you need to obtain the UDDI key of the entity and then insert the UDDI key to the above-mentioned URLs. For example, to get the information about the business entity with a UDDI key of `uddi:mydomain.com:39bb34ad-54e6-40e0-be2a-bd544e6772a4`, you need to:

- Replace *Host* and *Port* with those of the TIBCO ActiveMatrix Runtime UDDI Server, for example, **`localhost:58080`**.
- Replace *EntityType* with **`business`**;
- Replace *EntityKey* with **`uddi:mydomain.com:39bb34ad-54e6-40e0-be2a-bd544e6772a4`**.

The following example URLs can be used to get the information about the business entity.

- `http://localhost:58080/uddi/business/uddi:mydomain.com:39bb34ad-54e6-40e0-be2a-bd544e6772a4/xml`
- `http://localhost:58080/uddi/http_get?businessKey=uddi:mydomain.com:39bb34ad-54e6-40e0-be2a-bd544e6772a4.`

Getting Information about a Service Entity in XML Format

To get information about a service entity in XML format, you need to obtain the UDDI key of the entity and then insert the UDDI key to the above-mentioned URLs. For example, to get the information about the service entity with a UDDI key of `uddi:mydomain.com:0e050e6d-cb3d-480e-8f13-a08fec5d7861`, you need to:

- Replace *Host* and *Port* with those of the TIBCO ActiveMatrix Runtime UDDI Server, for example, **`localhost:58080`**.
- Replace *EntityType* with **`service`**;
- Replace *EntityKey* with **`uddi:mydomain.com:0e050e6d-cb3d-480e-8f13-a08fec5d7861`**.

The following example URLs can be used to get the information about the service entity.

- `http://localhost:58080/uddi/service/uddi:mydomain.com:0e050e6d-cb3d-480e-8f13-a08fec5d7861/xml`
- `http://localhost:58080/uddi/http_get?serviceKey=uddi:mydomain.com:0e050e6d-cb3d-480e-8f13-a08fec5d7861.`

Getting Information about a Binding Template in XML Format

To get information about a binding template entity in XML format, you need to obtain the UDDI key of the entity and then insert the UDDI key into the above-mentioned URLs. For example, to get the information about the binding template entity with a UDDI key of `uddi:mydomain.com:d2e8705f-3449-4fa7-888c-c9b85ef127d8`, you need to:

- Replace *Host* and *Port* with those of the TIBCO ActiveMatrix Runtime UDDI Server, for example, **`localhost:58080`**.
- Replace *EntityType* with **`binding`**;
- Replace *EntityKey* with **`uddi:mydomain.com:d2e8705f-3449-4fa7-888c-c9b85ef127d8`**.

The following example URLs can be used to get the information about the binding template entity.

- `http://localhost:58080/uddi/binding/uddi:mydomain.com:d2e8705f-3449-4fa7-888c-c9b85ef127d8/xml`
- `http://localhost:58080/uddi/http_get?bindingKey=uddi:mydomain.com:d2e8705f-3449-4fa7-888c-c9b85ef127d8.`

Getting Information about a tModel in XML Format

To get information about a tModel entity in XML format, you need to obtain the UDDI key of the entity and then insert the UDDI key to the above-mentioned URLs. For example, to get the information about the tModel entity with a UDDI key of `uddi:mydomain.com:bebc4299-3bc9-4975-9716-badfa9a2ce1c`, you need to:

- Replace *Host* and *Port* with those of the TIBCO ActiveMatrix Runtime UDDI Server, for example, **`localhost:58080`**.
- Replace *EntityType* with **`tModel`**;
- Replace *EntityKey* with **`uddi:mydomain.com:bebc4299-3bc9-4975-9716-badfa9a2ce1c`**.

The following example URLs can be used to get the information about the binding template entity.

- `http://localhost:58080/uddi/tModel/uddi:mydomain.com:bebc4299-3bc9-4975-9716-badfa9a2ce1c/xml`
- `http://localhost:58080/uddi/http_get?tModelKey=uddi:mydomain.com:bebc4299-3bc9-4975-9716-badfa9a2ce1c.`

Chapter 6 Integration with Other TIBCO Products

This chapter gives overview information on how to access TIBCO ActiveMatrix Runtime UDDI Server through other TIBCO products with the UDDI Registry function built in. This includes TIBCO Business Studio, TIBCO ActiveMatrix Administrator, TIBCO ActiveMatrix BusinessWorks, and TIBCO ActiveMatrix Policy Manager.

Topics

Integration with TIBCO ActiveMatrix BusinessWorks

In TIBCO ActiveMatrix BusinessWorks, you can search services published in TIBCO ActiveMatrix Runtime UDDI Server, import WSDL documents to your projects, deploy your projects through TIBCO Administrator and publish them to UDDI registries using TIBCO ActiveMatrix BusinessWorks plug-in for TIBCO Administration UDDI functionality.

To achieve this, a connection between TIBCO ActiveMatrix Runtime UDDI Server and TIBCO ActiveMatrix BusinessWorks is required. The UDDI Servers module allows you to define connections to TIBCO ActiveMatrix Runtime UDDI Server and view the web services contained in TIBCO ActiveMatrix Runtime UDDI Server. If you have been granted access to publish your own web services, you can also use the UDDI Servers module to publish information about your business and the web services you offer.

TIBCO ActiveMatrix BusinessWorks has two plug-ins: TIBCO ActiveMatrix BusinessWorks Designer plug-in and TIBCO ActiveMatrix BusinessWorks Administrator plug-in.

In TIBCO ActiveMatrix BusinessWorks, the design time operations, such as searching for services published in TIBCO ActiveMatrix Runtime UDDI Server and importing WSDL documents, are carried out by the TIBCO ActiveMatrix BusinessWorks Designer plug-in. The deploy operations, such as deploying projects, are carried out by TIBCO ActiveMatrix BusinessWorks Administrator plug-ins.

In TIBCO Designer or TIBCO Administrator, the Publication URL and Inquiry URL need to be provided:

- Publication URL: `http://`*Host*`:`*Port*`/uddi/services/publication`
- Inquiry URL: `http://`*Host*`:`*Port*`/uddi/services/inquiry`

For more information about accessing TIBCO ActiveMatrix Runtime UDDI Server through TIBCO ActiveMatrix BusinessWorks, see *TIBCO ActiveMatrix BusinessWorks Administration*, UDDI Servers Module, Chapter 7.

- Because UDDI V2 does not specify authentication, in order to use TIBCO ActiveMatrix BusinessWorks with TIBCO ActiveMatrix Runtime UDDI Server, you need to set inquiryAuthInfo to **`Ignored`** (the default) in TIBCO ActiveMatrix Runtime UDDI Server Administration Console. For more information, see *TIBCO ActiveMatrix Runtime UDDI Server Administrator's Guide*.
- When you search for services in TIBCO ActiveMatrix BusinessWorks, only those that contain at least one binding template with a tModel instance bound to it are listed.
- When you search for businesses in TIBCO ActiveMatrix BusinessWorks, only those that contain at least one service are listed. Note that each service must also contain binding templates with tModel instances bound to them.

Integration with TIBCO Administrator

In TIBCO Administrator, you can add UDDI registries and publish business and web service information to UDDI registries. This is achieved through the UDDI Servers module.

The UDDI Servers module allows you to define connections to UDDI servers and view the web services contained in the servers. If you have been granted access to publish your own web services, you can also use the UDDI Servers module to publish information about your business and the web services you offer.

For more information, see *TIBCO Administrator User's Guide*, Introduction, Chapter 1.

Integration with TIBCO Business Studio

In TIBCO Business Studio, you can search for services published in TIBCO ActiveMatrix Runtime UDDI Server and import services to your projects.

For more information, see Chapter 1 in *TIBCO ActiveMatrix Composite Editor User's Guide*.

Integration with TIBCO ActiveMatrix Administrator

TIBCO ActiveMatrix Administrator allows you to publish, unpublish, and synchronize the publish state of a service. To access a Runtime UDDI Server through TIBCO ActiveMatrix Administrator, you need first to add the TIBCO ActiveMatrix Runtime UDDI Server to TIBCO ActiveMatrix Administrator. With a Runtime UDDI Server added in TIBCO ActiveMatrix Administrator, you can perform the following operations:

- Enable/disable the TIBCO ActiveMatrix Runtime UDDI Server;
- Publish a service to the TIBCO ActiveMatrix Runtime UDDI Server;
- Remove a published service from the TIBCO ActiveMatrix Runtime UDDI Server.

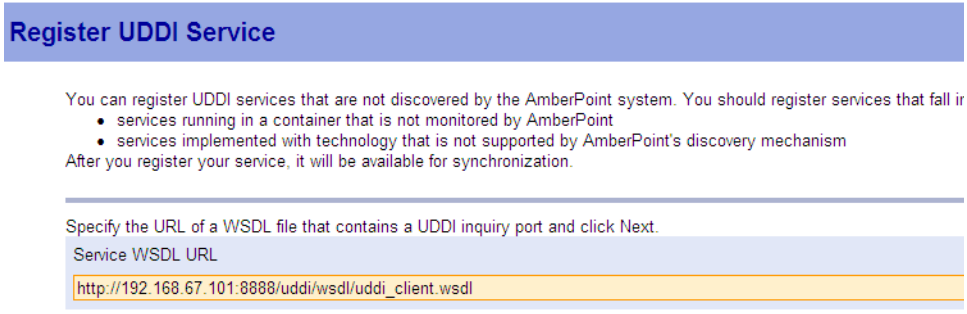
For more information, see the *TIBCO ActiveMatrix Administration* documentation.

Integration with TIBCO ActiveMatrix Policy Manager

TIBCO ActiveMatrix Runtime UDDI Server can interact with TIBCO ActiveMatrix Policy Manager to maintain public information about available services, endpoints, policies, and related resources.

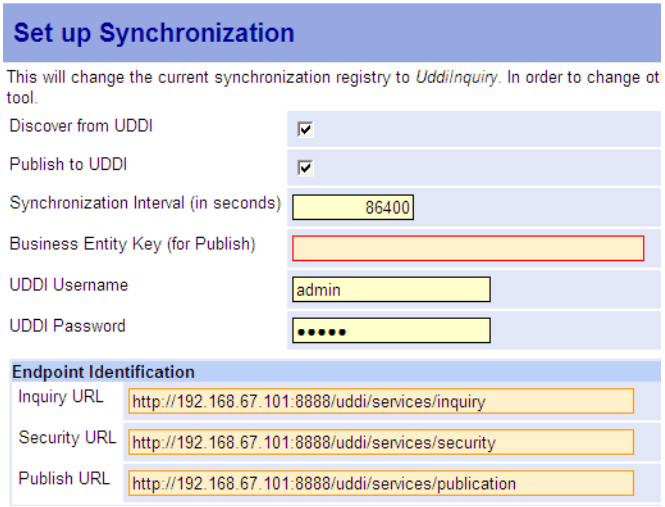
To enable TIBCO ActiveMatrix Policy Manager to interact with TIBCO ActiveMatrix Runtime UDDI Server, you need first to register the UDDI service in TIBCO ActiveMatrix Policy Manager, as shown in the following figure. (Remember to use your own Host and port settings in the Service WSDL URL.)

Figure 23 Register the UDDI Service



After registering the UDDI service, you need to set up synchronization and modify the Endpoint Identification settings manually, as shown in the following figure. (Remember to use your own Inquiry URL, Security URL, and Publish URL for Endpoint Identification settings.)

Figure 24 Set up Synchronization



For services hosted in TIBCO ActiveMatrix Serviced Grid, you can apply a policy directly to the services. When UDDI registry integration is enabled, TIBCO ActiveMatrix Policy Manager publishes the original service endpoint with a policy attachment for the tModel in the UDDI registry.

For the services hosted in TIBCO ActiveMatrix Service Grid with a policy attachment, it is recommended that you publish the services and policy attachments from the policy manager console perspective plug-in in TIBCO ActiveMatrix Administrator, not from the Monitor&manage perspective.

When applying policy to standalone services, a policy agent is needed. If the original service endpoint is already published in the UDDI registry, after the policy agent is started, TIBCO ActiveMatrix Policy Manage will publish an additional proxy agent endpoint in the UDDI registry as well as attach the policy attachment tModel to the service. In this case, the consumer of the service should discover and use the proxy agent endpoint from the UDDI registry.

TIBCO ActiveMatrix Runtime UDDI Server supports TIBCO ActiveMatrix Policy Manager version 3.0.0.

For more information on the interaction between TIBCO ActiveMatrix Runtime UDDI Server and TIBCO ActiveMatrix Policy Manager, see Chapter 7 in *TIBCO ActiveMatrix Policy Manager User's Guide*.

Chapter 7 Using Category Systems

This chapter describes how to use the two types of categories: checked category and unchecked category.

Topics

Introduction

The category system enables you to search UDDI entities by category. In TIBCO ActiveMatrix Runtime UDDI Server, a category is described as a tModel. TIBCO ActiveMatrix Runtime UDDI Server comes with a set of pre-loaded canonical tModels.

There are two types of categories in TIBCO ActiveMatrix Runtime UDDI Server: the checked category and the unchecked category.

Checked Category

When you attempt to publish an entity that is to be categorized as checked, the values are validated by a registered web service before the entity is created. If the values are invalid, the entity will be rejected by the UDDI registry.

Values can be validated in one of the following two ways.

- Values are cached by TIBCO ActiveMatrix Runtime UDDI Server

  In this case, valid values are cached in the memory. The category publisher must provide a web service with the `get_allValidValues` API and the tModel of the category must be categorized as *cacheable*. For information about the `get_allValidValues` API, see Section 5.6.3 get_allValidValues in *UDDI Version 3.0.2* at http://www.uddi.org/pubs/uddi_v3.htm#_Toc85908144.

  This type of validation is usually for categories that can be cached by the UDDI node. This is because their value sets are pre-defined and fixed.

- Values are not cached

  If a category requires contextual checking, each reference to the category will be validated by an external web service. The category publisher must provide a web service with the `validate_values` API and the tModel of the category must not be categorized as *cacheable*. For information about the `validate_values` API, see Section 5.6.2 validate_values in *UDDI Version 3.0.2* at http://uddi.org/pubs/uddi-v3.0.2-20041019.htm#_Toc85908143.

  This type of validation is usually for the references to the value sets required for contextual checking.

In both the cached and not cached cases, the tModel should be categorized with the *validatedby* category so that TIBCO ActiveMatrix Runtime UDDI Server can validate the values by calling corresponding web services.

For more information about checked categories and the related examples, see Using Checked Categories on page 108. The required files for running category system examples are in *UDDI_HOME*`/sample/data/taxonomy`.

Unchecked Category

For a category of this type, the reference is not checked. You can use any value (typically in the keyValue attribute of the keyedReference element) for the category.

An unchecked category can have an unlimited amount of values.

For more information about unchecked categories and the related examples, see Using Unchecked Categories on page 112. The required files for running category examples are in *UDDI_HOME*`/sample/data/taxonomy`.

Using Checked Categories

This section describes how to use checked categories.

Publishing a Checked Category

To publish a checked category:

1. Deploy a validation web service that provides the `get_allValidValues` API (for categories with their values for validation cached) or `the validate_values` API (for categories with their values for validation not cached).
2. Publish a tModel for the checked category. The tModel must be categorized by the built-in checked category. In addition, if the checked category is cacheable, the checked category must also be categorized by the built-in cacheable category.
3. Publish a binding template for the above-mentioned web service in the same UDDI registry as the tModel belongs to, with its technical fingerprint referencing the tModel whose key is `uddi-org:valueSetCaching_v3` (for the cacheable category) or `uddi-org:valueSetValidation_v3` (for the non-cacheable category) and the tModel published in step 2.
4. Update the above published tModel to have it point to the above binding template through the category whose UDDI key is `uddi-org:validatedBy` so that all the references to the value set of this category will be checked.

Checked Category Examples

There are two ways to do validation for a checked category. The following examples demonstrate how to use a checked category based on different validation types.

Using a Cached Category

This example demonstrates how to use a customized web service to check if a value belongs to a checked and cacheable category. The value validated in the example is referenced by the keyValue in a keyedReference element of a categoryBag element. The cached category is referenced by the tModelKey in the keyedReference element of the categoryBag element.

- The valid version numbers are defined in the `WEB-INF/classes/config.properties` file wrapped in the war file.
- The validation service in this example only accepts these version numbers: v1, v2, and v3.
- The web service used in this example is implemented using JAVA API for XML Web Services (JAX-WS).

To run the example:

1. Download the `JAX-WS 2.1.4` project from https://jax-ws.dev.java.net and install it. Copy the `lib` folder from the JAS-WS installation directory to the following directory:

   UDDI_HOME`/sample/data/taxonomy/checked-cachedvalue`

2. Open a command line window and change the path to *UDDI_HOME*`/sample/data/taxonomy/checked-cachedvalue`.
3. Build the `cachedValueValidationWebservice.war` file in the `war` directory by executing the following command:

   — `ant dist`

4. Deploy `cachedValueValidationService.war` in a web container.

   Replace the string `http://localhost:8080/` in the `publish_binding.xml` file with `http://`*Host*`:`*Port*`/`, where *Host* is the host name or IP address of the web container; *Port* is the web container port.

5. To publish the binding template, execute one of the following commands by using the data file `publish_binding.xml` file under the *UDDI_HOME*`/sample/data/taxonomy/checked-cachedvalue` directory.

   — `ant -f uddiant.xml save_binding`

   — `uddiant save_binding`

6. To publish the tModel, execute one of the following commands by using the data file `publish_tModel.xml` under the *UDDI_HOME*`/sample/data/taxonomy/checked-cachedvalue` directory.

   — `ant -f uddiant.xml save_tModel`

   — `uddiant save_tModel`

7. To publish a valid service, execute one of the following commands by using the data file `publish_service_data_valid.xml` under the *UDDI_HOME*`/sample/data/taxonomy/checked-cachedvalue` directory.

   — `ant -f uddiant.xml save_service`

   — `uddiant save_service`

8. To publish an invalid service, execute one of the following commands by using the data file `publish_service_data_invalid.xml` under the *UDDI_HOME*`/sample/data/taxonomy/checked-cachedvalue` directory.

   — `ant -f uddiant.xml save_service`

   — `uddiant save_serviceuddiant save_service`

The service will be rejected and an error message will appear.

You need to replace the domain name part of the keys in the sample data files with your node name used in the UDDI server.

Using an Uncached Category

This example demonstrates how to use a customized web service to check if a value belongs to a checked and uncached category. The value validated in the example is referenced by the keyValue in a keyedReference element of a categoryBag element. The uncached category is referenced by the tModelKey in the keyedReference element of the categoryBag element.

- In this example, only integers that are greater than or equal to 0 can pass the validation.
- The web service used in this example is implemented using JAVA API for XML Web Services (JAX-WS).

To run the example:

1. Download the `JAX-WS 2.1.4` project from https://jax-ws.dev.java.net and install it. Copy the `lib` folder from the JAS-WS installation directory to the following directory:

   UDDI_HOME`/sample/data/taxonomy/checked-uncachedvalue`

2. Open a command line window and change the path to *UDDI_HOME*/sample/data/taxonomy/checked-uncachedvalue.
3. Build the uncachedValueValidationWebservice.war file in the war directory by executing the following command:
   - ant dist
4. Deploy uncachedValueValidationService.war in a web container.

   Replace the string http://localhost:8080/ in the publish_binding.xml file with http://*Host*:*Port*/, where *host* is the host name or IP address of the web container; *Port* is the web container port.
5. To publish the binding template, execute one of the following commands by using the data file publish_binding.xml under the *UDDI_HOME*/sample/data/taxonomy/checked-uncachedvalue directory.
   - ant -f uddiant.xml save_binding
   - uddiant save_binding
6. To publish the tModel, execute one of the following commands by using the data file publish_tModel.xml under the *UDDI_HOME*/sample/data/taxonomy/checked-uncachedvalue directory.
   - ant -f uddiant.xml save_tModel
   - uddiant save_tModel
7. To publish a valid service, execute one of the following commands by using the data file publish_service_valid.xml under the *UDDI_HOME*/sample/data/taxonomy/checked-uncachedvalue directory.
   - ant -f uddiant.xml save_service
   - uddiant save_service
8. To publish an invalid service, execute one of the following commands by using the data file publish_service_invalid.xml under the *UDDI_HOME*/sample/data/taxonomy/checked-uncachedvalue directory.
   - ant -f uddiant.xml save_service
   - uddiant save_service

The service will be rejected and an error message will appear.

You need to replace the domain name part of the keys in the sample data files with your node name used in the UDDI server.

Using Unchecked Categories

This section describes how to use unchecked categories.

Publishing an Unchecked Category

There are two ways to publish an unchecked category:

- Publish a tModel to describe the category

  This is the formal way to publish an unchecked category. You can publish a tModel by executing a save_tModel task. For the procedures for executing a save_tModel task, see save_tModel on page 68.

- Use the built-in keyword category

  This is an informal way to publish an unchecked category. For a general keyword category, if you want to publish an entity, it can be categorized as the following:

```
<categoryBag>
   <keyedReference
     tModelKey="uddi:uddi.org:categorization:general_keywords"
     keyName="KeyName"
     keyValue="KeyValue"/>
   <keyedReference
     tModelKey="tModelKey"
     keyName="KeyName"/>
</categoryBag>
```

Note that `keyName` here is semantical and must not be empty. The required `keyName` identifies a particular value set and the `keyValue` specifies the value within that value set. The keyNames *SHOULD* be in the form of Uniform Resource Names (URN) so as to avoid name collisions, for example, *uddi:mydomain.com:demo*.

An Unchecked Category Example

The following example is about a version control system using an unchecked category.

To run the example:

1. Open a command line window and change the path to *UDDI_HOME*`/sample/data/taxonomy`

2. To publish the version control category system, execute one of the following commands by using the data file `publish_taxonomy_data.xml` under the *UDDI_HOME*`/sample/data/taxonomy` directory.

   — `ant -f uddiant.xml save_tModel`

   — `uddiant save_tModel`

3. To publish a category group system, execute one of the following commands by using the data file `publish_taxonomy_group_data.xml` (a category group contains a list of related categories) under the *UDDI_HOME*`/sample/data/taxonomy` directory.

   — `ant -f uddiant.xml save_tModel`

   — `uddiant save_tModel`

4. To check the category published in step 2 or the category group published in step 3, execute one of the following commands by using the tModel key in the corresponding file (that is, `publish_taxonomy_data.xml` or `publish_taxonomy_group_data.xml`).

   — `ant -f uddiant.xml get_tModelDetail`

   — `uddiant get_tModelDetail`

5. To publish a BusinessEntity, execute one of the following commands by using the data file `publish_business_data.xml` under the *UDDI_HOME*`/sample/data/taxonomy` directory.

   — `ant -f uddiant.xml save_business`

   — `uddiant save_business`

6. To publish BusinessServices in the BusinessEntity published in step 5, execute one of the following commands by using the data file `publish_service_data.xml` file under the *UDDI_HOME*`/sample/data/taxonomy` directory.

   — `ant -f uddiant.xml save_service`

   — `uddiant save_service`

   The BusinessServices to be published will be categorized using the categories published in step 2 and step 3.

7. To find the BusinessServices in the category published in step 2 or the category group published in step 3, execute one of the following commands.

   — `ant -f uddiant.xml find_service`

   — `uddiant find_service`

The inquiry can be performed in the following modes:

— **Interactive Mode**

Inquiries are performed based on the `keyedReference` elements.

— **Silent Mode**

A data file is required and inquiries can be performed based on the `keyedReference` or `keyedReferenceGroup` elements.

The following three files in the *UDDI_HOME*`/sample/data/taxonomy` directory are for this mode:

- `find_service_by_any_version.xml`
- `find_service_by_one_version.xml`
- `find_service_by_versions.xml`

8. To delete a category, execute one of the following commands.

— `ant -f uddiant.xml delete_tModel`

— `uddiant delete_tModel`

This command does not physically delete a category. It only marks the category as `deleted`.

The category key is the value of the tModelKey, such as the one defined in the `publish_taxonomy_data.xml` file.

The category value is the value of the keyValue defined in the `keyedReference` element, such as the one in the `publish_service_data.xml` file.

Appendix A UDDI Key Partition and Key Generator tModel

This appendix explains how UDDI keys are allocated.

Topics

UDDI Key Partition and UDDI Key Generator

A UDDI registry node can assign UDDI keys to entities that are created with the UDDI keys not provided by the publishers. To avoid UDDI key conflicts, UDDI key spaces are divided into non-overlapping, hierarchically arranged UDDI key partitions (hereinafter referred to as a partition). A UDDI node can only assign UDDI keys that are in specific partitions for objects created with the UDDI key not provided by publishers. In addition, according to UDDI registry policies, a publisher can also assign UDDI keys that are within specific partitions.

To assign UDDI keys to objects, a publisher must own the corresponding key generator tModel (a special kind of tModel). The key generator tModel contains a key generator key that identifies a partition. The publisher can only assign UDDI keys in the partition.

Typically, a publisher gets the ownership of a key generator tModel by publishing the tModel in question. A publisher can also get the ownership in other ways, for example, by having another publisher transfer the ownership.

You can assign UDDI keys for objects you are creating. Note that if the UDDI key you assigned to a newly created object is that of an existing object, the new object will overwrite the existing one.

Assigning Valid UDDI Keys to Objects in TIBCO ActiveMatrix Runtime UDDI Server

As mentioned in the previous section, only the UDDI keys belonging to partitions identified by key generator tModels that are already published in the system can be assigned to objects. So you need to know whether the corresponding key generator tModel is published in the system before assigning a UDDI key to an object.

You can find the key generator tModels already published in the system by performing an advanced search with the following settings:

- Object type: **`tModel`**
- Object name: **`:keygenerator`**
- Category: **`uddi:uddi.org:categorization:types`**
- Key name: **`uddi-org:types:keyGenerator`**
- Key value: **`keyGenerator`**

In TIBCO ActiveMatrix Runtime UDDI Server, if the key generator tModel with the UDDI key of `uddi:`*Yourspace*`:keygenerator` already exists, you can assign any UDDI keys prefixed with `uddi:`*Yourspace*`:` to objects to be published. You can:

- Publish a key generator tModel with its UDDI key being `uddi:`*Yourspace*`:it:keygenerator`.
- Publish an object with its UDDI key being `uddi:`*Yourspace*`:it:business`.
- Publish objects with their UDDI keys being `uddi:`*Yourspace*`:it:`*XXX*, `uddi:`*Yourspace*`:it:`*XXX*`.`*yyy*, and so on.

Currently, you can only publish key generator tModels through CLI. That is, you cannot publish key generator tModels through UDDI Service Console.

A Sample Data File for Publishing a Key Generator tModel

A key generator tModel is a tModel whose key is ended with the substring `:keygenerator`. Key generator tModels are categorized by the canonical tModel `uddi:uddi.org:categorization:types`.

The following is a sample data file for adding a tModel with the key of `keygenerator`.

```
<urn:tModelDetail xmlns:urn="urn:uddi-org:api_v3"
   xmlns="urn:uddi-org:api_v3">
```

```
   <tModel tModelKey="uddi:yourname:keygenerator">
      <name>your name keyGenerator</name>
      <description>your name key generator that allows to publish
       uddi entity with keys that starts with uddi:yourname:
       prefix</description>
      <categoryBag>
         <keyedReference tModelKey=
             "uddi:uddi.org:categorization:types"
             keyName="uddi-org:types:keyGenerator"
             keyValue="keyGenerator"/>
      </categoryBag>
   </tModel>
</urn:tModelDetail>
```

Following is a sample data file for searching for all the key generator tModels published.

```
<urn:find_tModel xmlns:urn="urn:uddi-org:api_v3"
      xmlns="urn:uddi-org:api_v3">
      <findQualifiers>
        <findQualifier>approximateMatch</findQualifier>
      </findQualifiers>
      <name>%:keyGenerator</name>
      <categoryBag>
         <keyedReference tModelKey=
              "uddi:uddi.org:categorization:types"
               keyName="uddi-org:types:keyGenerator"
               keyValue="keyGenerator"/>
      </categoryBag>
</urn:find_tModel>
```

Index

O

P

S

T

U