

TIBCO Adapter™ for COM

User's Guide

*Software Release 5.3
September 2005*

Important Information

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

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE TIBCO ADAPTER FOR COM USER'S GUIDE). 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, Information Bus, The Power of Now, TIBCO Adapter, TIBCO Rendezvous, TIBCO Administrator, TIBCO IntegrationManager, TIBCO Designer, TIBCO Hawk and TIBCO Enterprise are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

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

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

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

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

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

Copyright © 1999-2005 TIBCO Software Inc. ALL RIGHTS RESERVED.

TIBCO Software Inc. Confidential Information

Contents

Preface	ix
Related Documentation	x
TIBCO Product Documentation	x
Other TIBCO Product Documentation	x
How to Contact TIBCO Customer Support	xiii
 Chapter 1 Concepts	 1
Adapter Overview	2
Components	2
Adapter Features	4
Adapter Services	7
Publication Service	9
Request Response Invocation Service	10
Subscription Service	11
Request Response Service	11
Recommended Deployment Architecture	13
 Chapter 2 Installation	 15
Preparing your Environment for Installation	16
Operating System Requirements	16
Pre-Installation Worksheet	18
Adapter Machine Information	18
Installer Overview	19
Upgrading an Adapter	19
Uninstalling the Adapter	20
Installation Registry	21
Installation History	21
Adapter Components and Compatible Software	22
Adapter Components	22
Required and Optional TIBCO Products	22
Installing on Microsoft Windows	24
Installing on Microsoft Windows 2000 and 2003 Terminal Server	25
Installing the Adapter on Microsoft Windows	26
Combining Options	27
Post-Installation Tasks	27

Installation FAQs and Troubleshooting	29
Frequently Asked Questions	29
Speeding Up Installation	30
Running Out of Disk Space	32
Configuring TIBCO Hawk.	32
Chapter 3 Getting Started	35
Prerequisites	36
Overview	36
Permissions to Access Repository Server.	36
Other Examples	36
Create the Project	37
Create the COM Components.	39
Configure the Adapter	40
Configure the Publication Service.	43
Configure the Subscription Service.	44
Convert the Project to a Repository File	45
Deploy the Project and Start the Adapter	46
Chapter 4 Adapter Instance Options	47
Overview	48
Configuration Tasks	48
Adapter Instance Fields	49
Configuration Tab	49
Run-time Connection Tab	51
Adapter Services Tab.	52
General Tab	53
Logging Tab	54
Startup Tab	55
Monitoring Tab	56
Adapter Services.	58
Import Schema Tab	58
Publication Service Fields.	61
Configuration Tab.	61
Transport Tab.	62
SchemaView Tab	65
Schema Tab.	65
Subscription Service Fields.	66
Configuration Tab.	66
Transport Tab.	67
SchemaView Tab	71

Schema Tab	71
Request-Response Service Fields	72
Configuration Tab	72
Transport Tab	73
SchemaView Tab	76
Schema Tab	76
Request-Response Invocation Service Fields	77
Configuration Tab	77
Transport Tab	78
SchemaView Tab	80
Schema Tab	81

Chapter 5 Deploying and Starting an Adapter Using TIBCO Administrator Enterprise Edition. . 83

Overview	84
Create an EAR File in TIBCO Designer.	85
Deploy the Project	86
Start or Stop the Adapter	87
Monitor the Adapter	88

Chapter 6 Deploying and Starting the Adapter Using TIBCO Administrator Repository Edition 89

Overview	90
Export the Project	91
Create a Properties File.	92
Set Properties in Properties File	93
Assign Security Options.	94
Start or Stop the Adapter	95
Start the Adapter	95
Stop the Adapter.	98
Monitor the Adapter	99
Adapter Properties File	100
Properties File Format	100
Predefined Properties.	100
Server Repository URL	104
TIBCO Rendezvous	104
HTTP and HTTPS	105
Defining a urlFile to be Accessed via HTTPS.	107
Local Project URL	109
Install the Adapter as a Service on Microsoft Windows.	111

Chapter 7 Advanced Topics	113
Overview	114
Using the Adapter with a Revision Control System	115
Configuring a TIBCO Hawk Session	116
Using Global Variables	118
Variable Specification	118
Predefined Global Variables	119
Setting Encoding Options	122
Using DCOM	123
Using COM+	124
Configuring Highly Available Instances	125
Configuring the Adapter for Load Balancing Using RVCMQ	126
Configuring Interceptor Component	126
Configuring Service Component	126
Configuring the Adapter for Load Balancing Using JMS	128
Exception Handling	129
Obtaining Extended Error Information	131
Erroneous Method Calls	132
Chapter 8 TIBCO Moniker and Interceptor Activation	133
Overview	134
Display String Format	134
Parts of the Display String	135
Using the TIBCO Moniker in Visual Basic	139
Using the TIBCO Moniker in VC++	140
Endpoints and COM Coclasses	141
Method Signatures Must Match	141
Request Response Service	141
Request Response Invocation Service	142
Publication and Subscription Services	143
Mapping COM and TIBCO ActiveEnterprise Data Types	144
Chapter 9 Demonstration Applications	147
Overview	148
Repository Instances	148
Demonstration Projects	148
C++ Demonstration Programs	150
operation	150
C++ Client Projects	151

RecordSet Example	152
Connection Object Example	153
C++ Troubleshooting	155
Visual Basic Demonstration Programs	157
VBOperationDemo	157
VBPubSubDemo	158
Active Server Page (ASP) Demonstration: ASPDemo.	161
Bridging between TIBCO Messaging Transport and BizTalk	163
Define the COM interface	163
Pass Messages to BizTalk	164
Pass BizTalk Messages to TIBCO Messaging Transport	171
MSMQ Demonstration Example	176
Architecture	176
Example Scenario	177
.NET Remoting Examples	179
Examples Scenario.	180
Appendix A Data Types	181
TIBCO Adapter for COM Data Types.	182
VARIANTs	184
SAFEARRAYs	185
User Defined Types (UDTs).	186
COM Data Types	186
ENUMs	186
ADO Recordset Data Object.	186
ADO Connection Object.	189
General Comments on Data Types	190
Appendix B Interoperability with Microsoft .NET Components	191
COM and .NET Interoperability	192
Calling .NET Components from a COM Client	192
Calling COM Components from a .NET Client	192
Adapter and .NET Component Interoperability	193
.NET and COM Interoperation with the Adapter.	193
Sample Modules	195
Appendix C Monitoring the Adapter Using TIBCO Hawk	197
Overview	198
Starting TIBCO Hawk Software	199
The Auto-Discovery Process	200

Invoking Microagent Methods 201

Available Microagents 204

 activateTraceRole() 207

 deactivateTraceRole() 208

 getAdapterServiceInformation() 209

 getComponents() 210

 getConfig() 211

 getConfigProperties() 212

 getHostInformation() 213

 getRvConfig() 214

 getStatus() 215

 getTraceSinks() 216

 _onUnsolicitedMsg() 217

 preRegisterListener() 218

 reviewLedger() 219

 setTraceSinks() 221

 stopApplicationInstance() 222

 unRegisterListener() 223

 getActivityStatistics() 224

 getActivityStatisticsBySchema() 225

 getActivityStatisticsByService() 226

 getQueueStatistics() 227

 getThreadStatistics() 228

 resetActivityStatistics() 229

 resetThreadStatistics() 230

Appendix D Trace Messages 231

Overview 232

Trace Message Fields 234

Status Messages 237

Index 295

Preface

This manual describes how to install, configure, and use TIBCO Adapter for COM. It explains in detail the concepts and features of the adapter, and helps you get easily started by providing a simple configuration exercise.

Topics

- *Related Documentation, page x*
- *How to Contact TIBCO Customer Support, page xiii*

Related Documentation

This section lists documentation resources you may find useful.

TIBCO Product Documentation

The following documents form the TIBCO Adapter for COM documentation set:

- *TIBCO Adapter Concepts* — Read this manual to gain an understanding of adapters in general that you can apply to the various tasks you may undertake.
- *TIBCO Adapter for COM User's Guide* — This manual explains concepts relating to the adapter and the application with which it interacts. Installation, configuration and deployment information is included in this manual.
- *TIBCO Adapter for COM Examples Guide* — This manual provides hands-on examples that demonstrate use of the adapter.
- *TIBCO Adapter for COM Release Notes* — Read this document for information about new features, deprecated features, and open and closed issues.
- *README for TIBCO Adapter for COM* — Read this document to check the current release number and see a summary of software and hardware requirements for installing and running the adapter.

Other TIBCO Product Documentation

You may find it useful to read the documentation for the following TIBCO products. Note that only books that relate to adapters are listed. Each of the books is available from the `doc` directory in the product's installation area.

- TIBCO ActiveEnterprise™ software:
 - *TIBCO ActiveEnterprise Concepts*
- TIBCO Designer™ software:
 - *TIBCO Designer User's Guide*
 - *TIBCO Designer Palette Reference*
 - *TIBCO Designer Release Notes*

- TIBCO Administrator™ software:
 - *TIBCO Administrator User's Guide*
 - *TIBCO Administrator Server Configuration Guide*
 - *TIBCO Administrator Release Notes*
- - TIBCO BusinessWorks software:
 - *TIBCO BusinessWorks Concepts*
 - *TIBCO BusinessWorks Quick Start*
 - *TIBCO BusinessWorks Process Design Guide*
 - *TIBCO BusinessWorks Palette Reference*
 - *TIBCO BusinessWorks Installation*
 - *TIBCO BusinessWorks Release Notes*
- TIBCO IntegrationManager™ software:
 - *TIBCO IntegrationManager Concepts*
 - *TIBCO IntegrationManager Administrator's Guide*
 - *TIBCO IntegrationManager Process Design Guide*
 - *TIBCO IntegrationManager Reference*
 - *TIBCO IntegrationManager Release Notes*
- TIBCO Rendezvous™ software:
 - *TIBCO Rendezvous Concepts*
 - *TIBCO Rendezvous Administration*
 - *TIBCO Rendezvous Configuration Tools*
- TIBCO Enterprise™ for JMS software:
 - *TIBCO Enterprise for JMS User's Guide*
 - *TIBCO Enterprise for JMS Release Notes*
- TIBCO Hawk™ software:
 - *TIBCO Hawk Installation and Configuration*
 - *TIBCO Hawk Administrator's Guide*
- TIBCO Adapter SDK
 - *TIBCO Adapter SDK Concepts*

- TIBCO Runtime Agent
 - *TIBCO Runtime Agent Release Notes*
 - *TIBCO Runtime Agent Administrator's Guide*
 - *TIBCO Runtime Agent Installation*
 - *TIBCO ActiveEnterprise Features, Migration and Compatibility*

How to Contact TIBCO Customer Support

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

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

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

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

<http://support.tibco.com>

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

Chapter 1 **Concepts**

This chapter introduces the TIBCO Adapter for COM product and provides information about its features, components, and concepts.

Topics

- *Adapter Overview, page 2*
- *Adapter Features, page 4*
- *Adapter Services, page 7*
- *Recommended Deployment Architecture, page 13*

Adapter Overview

TIBCO Adapter for COM integrates COM-aware applications with other TIBCO ActiveEnterprise-compliant applications. It enables client applications to use COM interfaces to publish documents or send operation requests over the TIBCO messaging transport. It also enables COM objects to be driven by documents or operation requests received over the TIBCO messaging transport.

For a general introduction to adapters and the services they provide, read *TIBCO Adapter Concepts*.

Components

TIBCO Adapter for COM has two main components: the adapter palette and the run-time adapter.

The adapter palette comprises an adapter-specific GUI that seamlessly integrates with TIBCO Designer. It allows you to configure the adapter, and will store the configuration in persistent storage. (You will require TIBCO Designer for this, which is installed as part of the TIBCO Runtime Agent installation. TIBCO Runtime Agent must be installed before installing the adapter).

The run-time adapter uses this configuration to pass and convert data to and from the applications. The run-time adapter consists of the following:

- The *interceptor* (adcomInterceptor.dll or adcomInterceptor.exe) intercepts COM method calls, marshals the call parameters into a message, and sends that message over a TIBCO messaging transport. The message can be an operation request or a published document.

The interceptor has two implementations:

- a .dll COM server. This is used whenever COM requires a component to be implemented inside a .dll. The .dll implementation of the interceptor is activated through the "tibco" moniker.
- a Microsoft Windows Service COM server. Once loaded, the Microsoft Windows Service remains running indefinitely. The Microsoft Windows Service implementation provides superior scalability and throughput. The interceptor provides full support for TIBCO Rendezvous Certified Message Delivery (RVCM), which requires that a Certified Messaging (CM) sender remain alive. This allows the CM listener to register with the sender and tell the sender it has received messages so the sender can remove those messages from its ledger. The Microsoft Windows Service implementation of the interceptor is activated through the "tibcos" moniker.

For more information on activating the interceptor component through monikers, see TIBCO Moniker and Interceptor Activation on page 133.

- The *service* (`adcomService.exe`) creates instances of component classes and then listens for messages for those instances, published in the TIBCO ActiveEnterprise format on a TIBCO messaging transport. When a message is received, the service translates it into the appropriate method call on the default interface of the instance. The TIBCO ActiveEnterprise message can be an operation request or simply a published document.

Adapter Features

This section provides a brief overview of the adapter's features. These are discussed in detail later in the manual.

An Easy-to-Use GUI The adapter provides its own design-time component, namely the adapter palette, which seamlessly integrates with TIBCO Designer. This easy-to-use interface allows you to quickly configure adapter-specific features. You can use it to enter, delete, and modify configuration information. Additionally, you can import metadata using the adapter palette.

Support for Internationalization The adapter provides internationalization support based on the Unicode support provided by the TIBCO Adapter SDK. Support is provided for string data only. Schema, metadata, error messages, and all other non-string information do not support extended character sets. The encoding used by the adapter is UTF16_LittleEndian.

Support for Recordset Data Type The adapter supports Microsoft ADODB Recordset pointers as method parameters.

Support for ADO Connection Data Type The adapter supports Microsoft ADODB Connection pointers as method parameters.

Microsoft .NET Framework Compliance The adapter works seamlessly with Microsoft .NET components and clients. You can call a .NET component from a COM client or vice versa, using the adapter. A test example suite is also provided with the adapter installation for reference.

Runtime Connection Management The adapter starts trying to establish connection with out-proc COM servers at periodic intervals once it loses the connection with the COM server.

Support for Dual Message Transports The adapter supports the following message transports:

- **TIBCO Rendezvous Transport** — This transport uses subject-based addressing to provide support for both multicast or broadcast and point-to-point communications. You can configure the delivery modes of the messages and the wire format used when you configure the adapter service.
- **JMS Transport** — TIBCO Enterprise for JMS must be installed to use the JMS transport. The JMS administration interfaces allow you to create and manage administered objects such as Connection Factories, Topics, and Queues. JMS clients can retrieve references to these objects by using Java Naming and Directory Interface (JNDI). Creating static administered objects allows clients to use these objects without having to implement the object within the client. When a JMS client starts, it performs a JNDI lookup for the connection factories that it needs. For details on JNDI, see the *TIBCO Enterprise for JMS*

User's Guide. You can configure the connection factory type and the delivery mode using TIBCO Designer, when you configure the adapter service.

Distributed Queue Support A distributed queue is a group of cooperating transport objects, each in a separate process. Each transport object is called a member. To balance the transmission load among servers, the adapter can use distributed queues for *one-of-n* delivery of messages to a group of servers. Each member of a distributed queue must listen for the same subject using the TIBCO Rendezvous Distributed Queue listener objects. Even though many members listen for each inbound message (or task), only one member processes the message. For details on distributed queues, see *TIBCO Rendezvous Concepts*.

In the queue mode within TIBCO Enterprise for JMS, each listener is a single receiver of a point-to-point message. However, the listeners can be configured as a set of receivers, each of which receives a fraction of the messages. For details on TIBCO Enterprise for JMS distributed queues, see the *TIBCO Enterprise for JMS User's Guide*.

Load balancing for the processing of TIBCO Rendezvous or JMS certified messages is supported using distributed queuing. The messages from TIBCO Rendezvous or TIBCO Enterprise for JMS are distributed equally among all instances that belong to the same group. This distributes the data load over several adapter instances. However, the order in which the data is sent to the application server is not guaranteed.

Support for Quality of Service The adapter supports the following levels of service for TIBCO Rendezvous wire formats:

- Reliable messaging
- Certified messaging
- Distributed queues

The adapter supports the following levels of service for the JMS wire formats:

- Durable
- Non-durable

Platform Interoperability The adapter allows COM-based applications to easily interoperate with applications running on other platforms. For example, a Microsoft Windows application can make method calls through a COM interface to publish documents that are transported over the TIBCO messaging transport and consumed by a listening process running on a UNIX machine. Alternatively, a RPC client running on a UNIX machine can send an operation request to a RPC Server running in the service component of TIBCO Adapter for COM and have this request converted into an invocation of a method on a COM object.

Scalability Since the adapter is multi threaded, a single message being processed by the adapter does not become a bottleneck for subsequent messages. Simultaneous processing of messages results in high throughput and performance benefits. The TIBCO Rendezvous Distributed Queue and TIBCO Enterprise for JMS queue protocols for messaging allow peak load balancing between different instances of the same adapter configuration (subscription and request-response services) and also offers fault tolerance. In fault tolerance, when one listener fails, other listeners take over.

In the service component of the adapter, scalability can be achieved through three mechanisms:

- Multiple service components can be run as Microsoft Windows services on a single machine.
- Multiple service components can be run as members of a distributed queue group.
- You can increase the number of threads used by each service component by setting the `numEventThreads` configuration parameter in TIBCO Designer. Published documents and operation requests are then received and processed on these threads in parallel.

A COM client can use multiple threads to invoke COM methods. The client:

- Spawns the desired number of threads.
- Creates new COM objects inside those threads.
- Starts calling methods on those objects from the new threads.

Each method call is intercepted and processed by the interceptor component on the thread on which the method was called, so the interceptor scales as additional threads are created by the client.



The `sample.dat` project distributed with the adapter provides an example of how to achieve multi-threaded interceptor operation in a COM client.

High Availability In the service component of the adapter, high availability is achieved by running multiple service components as members of a distributed queue group.

Adapter Services

The interceptor component of the adapter provides the following:

- Publication Service
- Request Response Invocation Service

The service component of the adapter provides the following:

- Subscription Service
- Request Response Service

Interceptor

TIBCO Adapter for COM allows you to use COM interfaces to send messages over a TIBCO messaging transport. You do not need to write complex code to manage a messaging session or to marshal your data into messages. Instead, you can develop a COM interface and then use TIBCO Designer to set certain configuration parameters for the interceptor. The client application then instantiates the COM object through a TIBCO moniker, obtaining an interface pointer, and invokes methods on this interface pointer. The use of the TIBCO moniker to activate the coclass causes the interface pointer returned to your client to be an *interceptor proxy*.



For more information on the TIBCO moniker and interceptor proxies, see TIBCO Moniker and Interceptor Activation on page 133.

This proxy intercepts all method invocations made by the client, marshals the input parameters into a TIBCO ActiveEnterprise message (operation request or document), and places the message on the TIBCO messaging transport. To know how to perform the marshalling, the proxy utilizes the type library of the coclass and the configuration information you entered in the repository.

- If a document is being published or a one-way operation request is being sent, the calling thread returns to the client immediately.
- If a two-way operation request is being sent, the calling thread goes to sleep until a reply is received over the TIBCO messaging transport. On receipt of the reply, the proxy wakes up, unmarshals the operation reply into the output parameters for the method call, and returns to the client.

The operation of the interceptor proxy is completely transparent to the client. As far as the client knows, it is merely making method calls on an arbitrary COM interface. In reality, these method calls are being intercepted and converted automatically into TIBCO ActiveEnterprise messages.

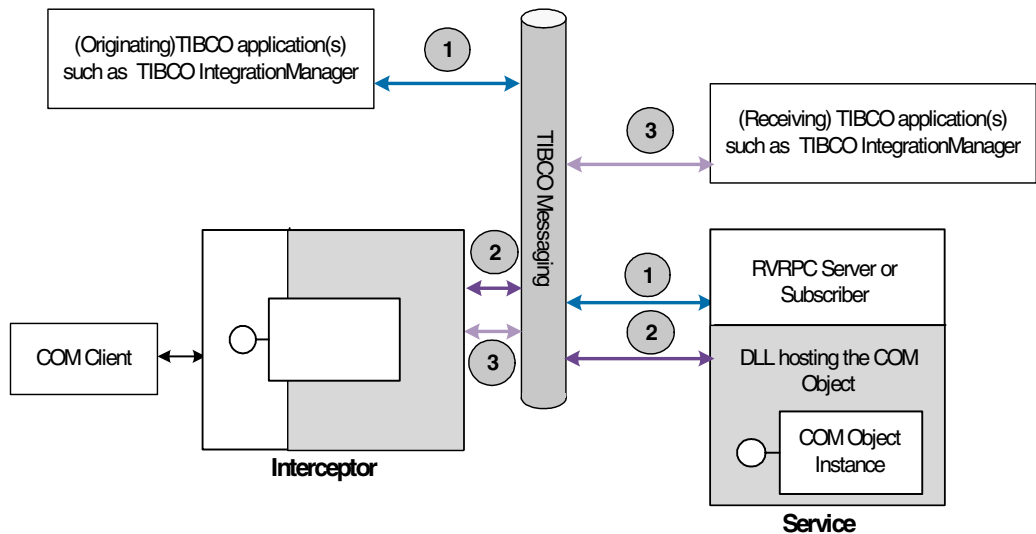
For example, you may configure TIBCO Adapter for COM so that it converts all calls to the method `INews::Politics` into documents to be published on the subject `NEWS.LOCAL.POLITICS`. Once you have mapped `INews::Politics` to a particular subject and defined other configuration preferences, you make calls to the `INews::Politics` method. These calls are intercepted and the call parameters are automatically marshaled into a document and published on the specified subject by the Interceptor component.

Service

TIBCO Adapter for COM also allows your COM components to be driven by messages flowing off a TIBCO messaging transport. Again, you do not need to write complex code to manage a messaging session or to unmarshal parameters from messages. You can use TIBCO Designer to map messages on a particular subject to a particular COM method. When you start the service component of TIBCO Adapter for COM, it starts listening for messages on the configured subject. When a message arrives, the service automatically unmarshals it and converts it into an invocation of the appropriate COM method call. When the call returns, if a two-way operation request was received, the service marshals the output parameters and the `HRESULT` from the stack into an operation reply and returns this reply to the client.

The following figure illustrates the interceptor and service.

Figure 1 Interaction between the Interceptor and Service Component

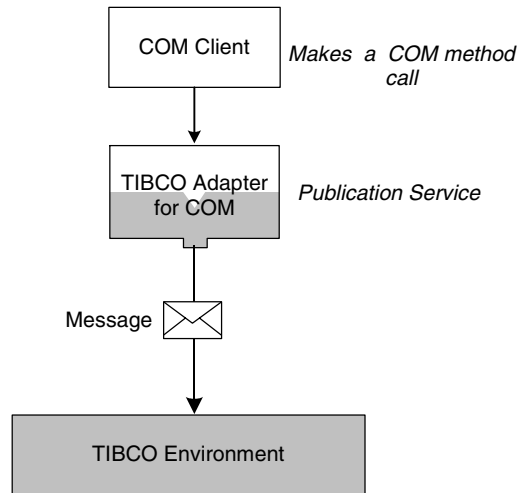


- 1-1: Messages originating from TIBCO applications are published to the TIBCO messaging transport, and are received by the service component of the adapter.
- 2-2: The COM client invokes the interceptor component of the adapter, which publishes the messages to the TIBCO messaging transport. These are received by the service component of the adapter.
- 3-3: The COM client invokes the interceptor component of the adapter, which publishes the messages to the TIBCO messaging transport. These are received by TIBCO applications such as TIBCO IntegrationManager.

Publication Service

The interceptor component of the adapter intercepts a method call from a COM client, converts it into a message, and sends the message to the TIBCO environment.

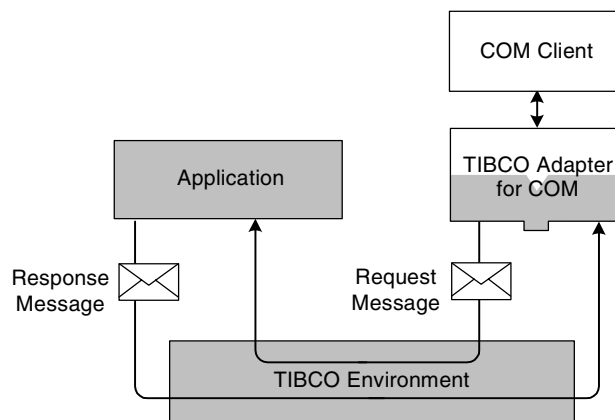
Figure 2 Typical Publication Service Flow



Request Response Invocation Service

The interceptor component of the adapter gets a request from a COM client and sends the request through the TIBCO environment. When a response is returned from the TIBCO environment, the adapter sends the response back to the COM client.

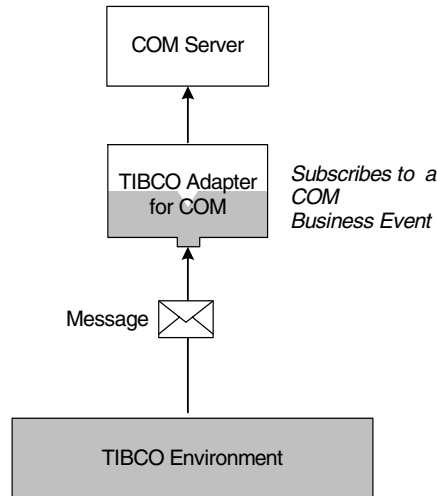
Figure 3 Typical Request Response Invocation Service Flow



Subscription Service

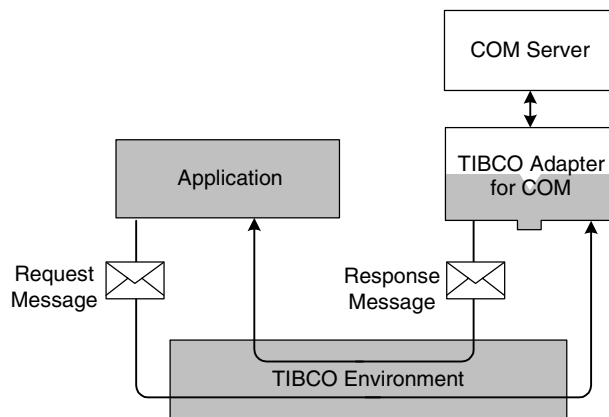
The service component of the adapter gets a message from the TIBCO environment and invokes the COM method on the COM server.

Figure 4 Typical Subscription Service Flow



Request Response Service

The service component of the adapter gets a request from the TIBCO environment and sends the request to a COM server. When a response is returned to the adapter from the COM server, the adapter sends the response to the TIBCO environment.

Figure 5 Typical Request Response Service Flow

The adapter supports request response scenarios with an RPC server. When the adapter receives a request, it takes the raw requested data, converts it into a formatted schema, and sends it to COM server by using a designated COM interface.

Recommended Deployment Architecture

The service component of the adapter can be deployed as a standalone executable or as a Microsoft Windows service. When running multiple instances of the service component as members of a distributed queue, these instances may be run on separate machines.

The interceptor component of the adapter can also be deployed as a standalone executable or as a Microsoft Windows service.

For details, see Chapter 5, Deploying and Starting an Adapter Using TIBCO Administrator Enterprise Edition, on page 83 and Chapter 6, Deploying and Starting the Adapter Using TIBCO Administrator Repository Edition, on page 89.

Chapter 2 **Installation**

This chapter explains how to install the adapter on Microsoft Windows and UNIX systems.



This software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. Please see Table 3 on page 24 for the availability of this software version on a specific operating system platform.

Topics

- *Preparing your Environment for Installation, page 16*
- *Pre-Installation Worksheet, page 18*
- *Installer Overview, page 19*
- *Installation Registry, page 21*
- *Adapter Components and Compatible Software, page 22*
- *Installing on Microsoft Windows, page 24*
- *Installation FAQs and Troubleshooting, page 29*

Preparing your Environment for Installation

The most time-consuming part of an adapter installation is the collection of environment information and parameters. This section helps you complete this process. It provides a check list of parameters you should obtain from various system administrators within your organization before installing the adapter.

Operating System Requirements

Obtain the following information from the administrator of the machine on which you plan to install the adapter:

System name:
System IP address:
Username and password to access the system and run the adapter:
Username:
Password:

Do you have the required credentials to run the installer?

- On Microsoft Windows, administrator privileges are required to install.
- Note that the TIBCO Runtime Agent must be installed prior to installing the adapter and the adapter installation always places files under the TIBCO root directory that was set when the TIBCO Runtime Agent was installed.

Is there enough space on that disk or partition to install the adapter? The adapter needs space in your temp area and the directory where it is installed.

- See Installation Registry on page 21 for details about temp folder space requirements on Microsoft Windows systems.
- See Table 3 on page 24 for Microsoft Windows installations

You must have write permissions to these directories to install the adapter. To run the adapter you must have permissions to access the project where adapter configuration is stored.

- Depending on whether TIBCO Administrator is used to set access permissions, you may need an account identified by Administrator. See the *TIBCO Administrator User's Guide* for details.

Determine how the adapter installation files are to be transferred to this system. The installation files can be downloaded from download.tibco.com (if you have an account setup to download). Do you plan to use FTP, NFS, HTTP, or install from a CD?

Pre-Installation Worksheet

Use this form to capture the information you will need to collect before starting installing the adapter

Adapter Machine Information

Field Name	Field Description	Field Value
Hostname (Example: adapter1.tibco.com)	Name of the machine on which the adapter is being installed.	
IP address (Example: 192.168.12.12)	IP address of the machine on which the adapter is being installed.	
User account (Example: administrator)	User account to be used for the installation.	
User domain (if Microsoft Windows) (Example: ENGR2)	Network domain to which the user belongs.	
User password		
Disk and path on which to install adapter (Example: C:\tibco)		
How will machine be accessed	<input type="checkbox"/> directly <input type="checkbox"/> terminal server <input type="checkbox"/> xterm <input type="checkbox"/> telnet <input type="checkbox"/> other:	
How will installation files be transferred to machine	<input type="checkbox"/> CD-drive <input type="checkbox"/> internet download <input type="checkbox"/> FTP to machine <input type="checkbox"/> network disk mounting	

Installer Overview

The installer allows you to run in different modes. Each mode is supported on all platforms.

- GUI mode
- Console mode
- Silent mode

GUI Mode

In GUI mode, the installer presents panels that allow you to make choices about product selection, product location, and so on. When you invoke the installer by double-clicking the icon, GUI mode is used.

Console Mode

Console mode allows you to run the installer from the command prompt or terminal window. This is useful if your machine does not have a Microsoft Windows environment.

Silent Mode

Silent mode either installs using default settings or uses a response file that was saved during an earlier installation. Silent mode installs without prompting you for information.

- If no response file has been recorded earlier and you invoke the installer with the `-silent` argument, the default installation parameters are used.
- If a response file exists, and the installer is started with `-options <responseFileName>` as an argument, the installer uses the values specified by the user when the response file was generated.

Upgrading an Adapter

Software from TIBCO uses three numbers to indicate whether the release is major, minor or a patch. For example, 5.0.0 indicates a major release, 5.1.0 indicates a minor release and 5.1.1 indicates a patch release. The installer for a patch release performs an automatic upgrade. For example, the installer automatically upgrades TIBCO Runtime Agent 5.0.0 to 5.0.1 by overwriting the contents of the 5.0 directory.

For a major and minor release, the installer prompts whether you wish to upgrade, and informs you if incompatible products are on your system. If you proceed, major or minor releases are installed under a new directory that is named using the major or minor release numbers.

For example, if you have installed the 5.0.0 release and are upgrading to a 5.1.0 minor release, it will be installed under the 5.1 directory. This allows both the 5.0 and 5.1 releases to coexist on the same machine.

If you are upgrading the adapter, or reinstalling a clean version of the software, you may uninstall the product first or allow the installer to perform the upgrade or reinstall.

Note that, if you are reinstalling over the same adapter version:

- You are not prompted to supply the installation location. The software is automatically reinstalled where the previous version was installed.
- If any files are currently locked (that is, in use), the installer marks the file for deletion in the install location. After installation, the installer prompts you to reboot your system. You must reboot before using the software.

Uninstalling the Adapter

The TIBCO Installation Manager allows you to:

- Display installed products
- Display product components
- Display product dependencies and references
- Uninstall products and all references

To start the program:

- Click **Start>TIBCO>TIBCO Installation Manager**

The TIBCO Installation Manager displays all TIBCO software installed on your machine. Details about the selected TIBCO product are displayed in the right rows. Click the **Uninstall** button to remove the selected product. Note that some products cannot be installed from the utility. Instructions about uninstalling these products are listed in the uninstall window (after clicking the **Uninstall** button).

Click the TIBCO Installation Manager help button for information about using the program.

Installation Registry

The installer maintains an installation registry. The registry location depends on platform. This section explains where the registry files are located. The files have vpd as a prefix, which stands for Vital Product Database. Note that the installer does not recognize TIBCO ActiveEnterprise 4.x products.



Do not edit, modify, rename, move, or remove any of the registry vpd files.

ActiveEnterprise 5.2 products maintain the installation registry in the `SystemDrive:\WINNT` directory. The following files represent the installation registry:

```
SystemDrive:\WINNT\vpd.properties
SystemDrive:\WINNT\vpd.properties.tibco.systemName
```

Installer Disk Space Requirements in Temporary Area

The entire package is extracted into a temp folder, typically `SystemDrive:\Temp` or `SystemDrive:\Documents and Settings\<user_name>\Local Settings\Temp`.

The installer requires 45MB of free space in the temp directory.

Installation History

The installer and uninstaller creates a file called `TIBCOInstallationHistory.xml` in the same location where the installation registry is created. Each time an installation and uninstallation is performed, entries are appended to the file,

```
SystemDrive:\WINNT\TIBCOInstallationHistory.xml
```

The file `TIBCOInstallationHistory.xml` therefore contains the record of all installation and uninstallation activities of all products, features and components.



Do not edit, modify, rename, move, or remove the `TIBCOInstallationHistory.xml` file.

Adapter Components and Compatible Software

You can install different adapter components on different machines. For example, you can run the runtime adapter on one machine and install the design-time components on another machine. This allows you to configure an adapter on one machine and run it on another.

Adapter Components

Table 1 describes the adapter components on the adapter installation package.

Table 1 TIBCO Adapter components

Component	
Run-time adapter	This process does the actual work of passing and converting data. Parameters of data exchanges are stored in projects created using the adapter palette.
Adapter palette	Adapter-specific GUI that is loaded in TIBCO Designer (see next section for details) at configuration time.

Required and Optional TIBCO Products

Depending on the tasks you wish to perform, you must install one or more other TIBCO products. The next table describes required and optional products and their purpose.

Table 2 Required and Optional TIBCO Products

Component	Purpose
TIBCO Runtime Agent 5.2.1	<p>Required. TIBCO Runtime Agent supplies a number of TIBCO and third-party libraries used by the adapter and other TIBCO products both at design-time and runtime. This includes, for example, TIBCO Rendezvous and TIBCO Designer software.</p> <p>You must install TIBCO Runtime Agent on each machine that hosts an adapter. TIBCO Runtime Agent must be installed before you install the adapter.</p>
TIBCO Designer 5.2.0	<p>Required. TIBCO Designer is an easy to use graphical user interface for design-time configuration of TIBCO adapters.</p> <p>TIBCO Designer is installed as part of the TIBCO Runtime Agent installation.</p>

Table 2 Required and Optional TIBCO Products

Component	Purpose
TIBCO Administrator 5.2.1	<p>Required. TIBCO Administrator is available in two editions, Repository Edition and Enterprise Edition. You must install either edition. The Repository Edition does not include the Application Management module. Enterprise Edition is required if you are using the adapter with TIBCO BusinessWorks. Administrator includes the following modules:</p> <ul style="list-style-type: none"> • User Management. Management of authentication, roles and users, that is, connecting roles (groups) and users to access control lists (ACLs). This includes security for server-based projects at design-time and for deployed applications at runtime. • Resource Management. Monitoring of machines and of all running applications in a TIBCO administration domain. Alerts can be created, for example, to notify an administrator if the number of processes or disk usage exceed a certain number. • Application Management. Uploading of Enterprise Archive (EAR) files, creation, configuration, deployment, and monitoring of applications. This console is also used to start and stop applications. <p>TIBCO Administrator is available as a separate installation and can be installed after installing the adapter.</p>
TIBCO BusinessWorks 5.2.1	<p>Optional. TIBCO BusinessWorks is a scalable, extensible, and easy to use integration platform that allows you to develop integration projects. TIBCO Adapters are typically part of integration projects created using BusinessWorks.</p> <p>TIBCO BusinessWorks is available as a separate installation and can be installed after installing the adapter.</p>
TIBCO Enterprise Message Service 4.2	<p>Optional. TIBCO Enterprise for JMS allows you to use the Java Messaging Services (JMS) as the message transport for your adapter.</p> <p>TIBCO Enterprise for JMS is available as a separate installation and can be installed after the adapter is installed.</p>
TIBCO Hawk 4.5.1	<p>Optional. TIBCO Hawk allows you to use the adapter's built-in Hawk microagents to monitor and manage the adapter.</p> <p>TIBCO Hawk is available as a separate installation and can be installed after the adapter is installed.</p>

Installing on Microsoft Windows

Before starting the installation procedure, review the topics in this section to determine that your system meets the basic requirements and that you have the prerequisite software installed.

The following is a list of prerequisites for installing the adapter on Microsoft Windows systems. See *Installer Disk Space Requirements in Temporary Area* on page 21 for additional disk space requirements.

Table 3 Supported platforms, package names, service packs and disk space for Microsoft Windows

Platform	Package Names	Service Pack	Hardware	Disk Space (MB)
Microsoft Windows 2000		Service Pack 3	x86	45
Microsoft Windows XP Professional	TIB_adcom-simple_5.3.0_w32.exe	Service Pack 1a	x86	45
Microsoft Windows Server 2003			x86	45



If you are using the Recordset feature or the ADO connection object feature, you will need version 2.5 of the Microsoft ADODB libraries.

TIBCO Runtime Agent Must be Installed Before the Adapter

Before you can install the adapter, you must install the TIBCO Runtime Agent. If you use the Typical installation, the installer places all libraries and other products required by the adapter into the TIBCO HOME directory.

During installation, the adapter installer checks for the availability of all dependent products in the target system. If any of the dependencies are not available, the installer will immediately exit. Otherwise installation will proceed.



This release of the adapter requires TIBCO Runtime Agent 5.2.1 Hotfix3.

Installer Account

You must have administrator privileges for the machine on which the adapter is installed.

If you do not have administrator privileges, the installer will exit. You must then log out of the system and log in as a user with the required privileges, or request your system administrator to assign the privileges to your account.

Installing from Network Drive

If you intend to install the product on a network drive, you must ensure that the account used for installation has permission to access the network drive.

Installing on Microsoft Windows 2000 and 2003 Terminal Server

There are two modes in the Microsoft Windows Terminal Server: *Execute* and *Install*. Users are logged on by default in the Execute mode, which allows them to run applications. To install an adapter so that everyone can use it, log on as administrator in Install mode. When the adapter is installed in the Install mode, the installation registry is maintained in *SystemDrive:\WINNT*.



Microsoft Windows Terminal Server must be running in remote admin mode, not application sharing mode. The adapter is not supported if it is installed on a machine that is using Microsoft Windows Terminal Server in the application sharing mode.

The best way to install the adapter on Microsoft Windows Terminal Server is to use the Add/Remove Programs control panel applet. This automatically sets your mode to Install during the installation and then back to Execute afterwards. Alternatively, you can manually change your mode to Install before starting the installation by typing the following at a command prompt:

```
C:\> change user /install
```

Change back to Execute mode after installation is complete by typing:

```
C:\> change user /execute
```

To check your current mode, type the following:

```
C:\> change user /query
```

Installing the Adapter on Microsoft Windows

You can either download the adapter package or install the package from a CD. The installer prompts you to accept the license agreement, then to choose to perform a typical install or custom install.

- A typical install has minimal prompts and installs standard components in default locations.
- A custom install prompts you to choose which components of the product suite to install and installs only those components.

The installer checks your system for the installation home directory that was established when TIBCO Runtime Agent was installed. The adapter is installed under the installation home directory.

Use one of the following modes to install the software.

Install Using GUI Mode

GUI Mode allows you input values in panels. Type the following at the command prompt:

```
TIB_adcom-simple_5.3.0_w32.exe
```

Install Using Console Mode

Console mode allows you to install the software in a non-Microsoft Windows environment. The installer will prompt you for values. Type the following at the command prompt:

```
TIB_adcom-simple_5.3.0_w32.exe -is:javaconsole -console
```

When running in console mode you can move through the installation process as described next:

```
Enter Key = Moves forward in the installer
2 = Goes back to previous screen
3 = Cancels the Wizard and exits the installation or uninstallation
4 = Redisplays the current screen
```

Install Using Silent Mode

Silent mode allows you to install the software without prompts. Type the following at the command prompt:

```
TIB_adcom-simple_5.3.0_w32.exe -silent
```


Install and Generate a Response File

You can generate a response file during installation which you can later use to invoke the installer with the selected values as default values (GUI mode) or as selected values (silent mode).

To install and generate a response file, type the following at the command prompt:

```
TIB_adcom-simple_5.3.0_w32.exe -options-record
                                C:\directory\

```



The response file does not record selections at the component level. It does record all other selections, for example, which products you wished to install.

Install Using a Response File

You can use a previously generated response file for installation. For non-silent modes, the response file determines the defaults that are presented. For silent mode, the response file determines what will be installed.

To install using a response file, type the following at the command prompt:

```
TIB_adcom-simple_5.3.0_w32.exe -options
                                C:\directory\

```

Combining Options

You can combine the different available options. For example, to install in silent mode using a response file, use:

```
TIB_adcom-simple_5.3.0_w32.exe -silent -options <responseFileName>
```

To install using Console mode and generate a response file, use:

```
TIB_adcom-simple_5.3.0_w32.exe -is:javaconsole -console
                                -options-record <responseFileName>
```

Post-Installation Tasks

- Verifying System Variables
- Running the Interceptor

Verifying System Variables

Ensure that the installer has set the system variable TIB_ICU_DATA to the <TRA_HOME>\config\g11n folder. If this variable has not been set, you must set it manually.

To set the system variable `TIB_ICU_DATA` manually:

1. Open the Environment Variables applet from the Control Panel.
2. Under System Variables, click **New**. The New System Variable dialog is displayed.
3. In the Variable Name field, type `TIB_ICU_DATA`.
4. In the Variable Value field, type the path to the `<TRA_HOME>\config\g11n` folder.

Running the Interceptor

To run the interceptor, you must run the `setenv.bat` file first so that the path is set. The installer installs the interceptor as:

- a COM dll

If the Interceptor is installed as a COM dll, it is dependent on the other dlls packaged with the adapter as well as on the TIBCO infrastructure products for its functioning. When the interceptor is used from within a Microsoft VisualBasic, Visual C++, or ASP code, it attempts to locate the dependent dlls. Therefore, you must update the system environment variables after the adapter installation has been successfully completed and restart the machine, as described next.

- a Microsoft Windows service.

However, you must update the system path with the path specified in the `setenv.bat` and restart the machine to use the interceptor as a Microsoft Windows service, as described next.

To update the system variable:

1. Include the following in the system variable `PATH`:
 - `<ADCOM_HOME>\hotfix\bin` and `<ADCOM_HOME>\hotfix\lib`
 - `<ADCOM_HOME>\bin` and `<ADCOM_HOME>\lib`
 - `<TRA_HOME>\bin` and `<TPCL_HOME>\bin`
2. Restart the machine.
3. Register the interceptor(.dll) from the `<ADCOM_HOME>\bin` directory.

Installation FAQs and Troubleshooting

This section lists some common errors along with their causes and solutions.

Frequently Asked Questions

Where is the installation log file located?

Install and uninstall log files are created in the TIBCO_HOME\log directory.

What should I do, if JVM crashes when I run the installer?

The installer first extracts the bundled JVM into a temporary area and then uses it to launch itself. If for some reason, the JVM crashes, you could still run the installer using another JVM, preferably JVM 1.3.1 or higher. The syntax is:

```
TIB_adcom-simple_5.3.0_w32.exe -is:javahome C:\j2sdk1.4.0
```

The javahome directory must contain bin/java.exe or bin/java.

The installer will use the externally supplied JRE to launch itself.

Will 5.1 installer recognize a 3.x or 4.x installation?

TIBCO products follow a three digit release numbering scheme:

Major.Minor.Maintenance

Product releases that differ in either Major or Minor numbers will be a separate installation, and will not recognize the old installation. In this case, 5.0 is a major release and hence will not recognize either 3.x or 4.x product installations.

What is uninst2 directory?

If the original uninstall directory is in use at uninstall time, it cannot be removed by the installer program. The installer then creates a second uninstall directory for the second installation. To remove the second installation, you must invoke the uninstall program from the second uninstall directory. The original uninstall directory can also be manually removed, if empty.

Speeding Up Installation

The installer for TIBCO 5.x products is Java-based and requires a Java Virtual Machine (JVM) to launch. If the installer can use a JVM already on your system, installation is faster and uses less disk space. Otherwise, the installer extracts JVM from the installation package. This section explains where the installer searches for JVM, and how you can add a different location to the search path.

Every product installer is bundled with the appropriate platform-specific JVM version. When launched, the installer first searches for the appropriate JVM version on the target system.

- If the installer finds the required JVM version, it uses that JVM to launch itself.
- If the installer does not find the required JVM version, it extracts the bundled JVM into a temporary space and uses that JVM to launch itself.

How the Installer Searches for JVM

The installer by default searches for an appropriate JVM version in a set of standard locations. These standard locations are platform specific and are listed in JVM Search Locations on page 31. On each platform, the installer searches for JVM 1.3.1 only.

Speeding the JVM Search

You can speed up the JVM search by taking one of these actions before you launch the installer:

- Set the environment variable `JAVA_HOME` or `JDKHOME` to the JVM home directory, for example:

Microsoft Windows	<ol style="list-style-type: none">1. Choose Start->Settings->Control Panel->System-Environment->System Variable or User Variables2. Add a <code>JAVA_HOME</code> variable and set its value <p>NOTE: On Microsoft Windows, setting <code>JAVA_HOME</code> in a command prompt will not be effective.</p>
-------------------	---

- Include a command line option `-is:javahome` when launching the installer. For example:

```
Microsoft      <exe> -a -is:javahome c:\java\JRE\1.3.1
Windows suite
installer
```

Benefits of Using Available JVM over Extracting Bundled JVM

- JVM search is quick. It is searched for in well defined locations. It can be sped up by setting an environment variable or by using a command line option.
- Extraction of bundled JVM is avoided, which saves both time and temporary disk space. Initial loading of the installer is faster.
- If the bundled JVM is extracted and used for launching the installer, then a copy of this JVM is installed for launching the uninstaller in the `_jvm` directory in the product location.

However, if JVM was resolved through JVM search, the uninstaller just points to this JVM and uses it later to launch itself. In this case, JVM is not installed. This is a major benefit in terms of disk space.

- If the JVM the uninstaller points to is removed at any time, you can still run the uninstaller using the `-is:javahome` option to point to a different JVM location.

JVM Search Locations

The following tables list the search locations for the Microsoft Windows platforms.



The installer only searches for JVM version 1.3.1. If a different JVM version exists in the system, or is passed in using the environment variable or command-line option, the installer ignores that JVM and extracts the bundled JVM.

Table 4 Microsoft Windows (JVM 1.3.1)

Location	Description
Environment	Set in Control Panel, not in command prompt JAVA_HOME JDKHOME

Table 4 Microsoft Windows (JVM 1.3.1)

Location	Description
Registry	\HKEY_LOCAL_MACHINE\SOFTWARE\JavaSoft\Java Runtime Environment\1.3\JavaHome \HKEY_LOCAL_MACHINE\SOFTWARE\JavaSoft\ Java DevelopmentKit\1.3\JavaHome
Directory	\Program Files\JavaSoft\JRE\1.3.1 \JavaSoft\JRE\1.3.1 \Java\JRE\1.3.1

Running Out of Disk Space

The installer calculates the disk space required in product home location, for the selected components. The calculation is done before the actual installation (copying of files to system) begins. The installer will proceed only if sufficient free disk space is available in product home location.

However, if disk space is consumed by another process while the installer is copying the files, and if the required disk space is thereby reduced, the installer may fail and will then give a failure message.

Solution

While performing installation, avoid running other processes that consume disk space in product home location.

Configuring TIBCO Hawk

Error

TIBCO Runtime Agent includes the TIBCO Hawk Agent only. If you install the full TIBCO Hawk package after installing TIBCO Runtime Agent and do not have a Java Runtime Environment (other then the TIBCO JRE) installed, the TIBCO Hawk Configuration tool is unable to determine the Java home location and the JVM executable. The TIBCO Hawk services will not start correctly and you will be unable to start the TIBCO Hawk Display.

Resolution

1. Start the TIBCO Hawk Configuration tool. For example, on Microsoft Windows:

```
Start>TIBCO>TIBCO Hawk>Hawk Configuration
```

2. Under the General tab, click **Advanced**.
3. In the Java Home Directory field, provide the path to Java. For example:
C:\tibco\jre\1.4.2
4. In the JVM Executable field, provide the JVM executable. For example:
java.exe

The services will start properly and the TIBCO Hawk Display will run.

Chapter 3

Getting Started

This chapter presents examples that demonstrate key adapter features. Work through these examples to get a hands-on understanding of how the adapter works.

Topics

- *Prerequisites, page 36*
- *Create the Project, page 37*
- *Create the COM Components, page 39*
- *Configure the Adapter, page 40*
- *Configure the Publication Service, page 43*
- *Configure the Subscription Service, page 44*
- *Convert the Project to a Repository File, page 45*
- *Deploy the Project and Start the Adapter, page 46*

Prerequisites

Before starting the configuration exercise, ensure that all required software has been installed and is operating correctly. For a list of required software, see the installation instructions in Chapter 2, *Installation*, on page 15.

You should know how to drag and drop icons in TIBCO Designer and be familiar with saving projects. If you are not familiar with these topics, refer to the *TIBCO Designer User's Guide*, which is available by clicking **Help>Designer Help** in TIBCO Designer.

Overview

This exercise assumes that you will use `sample` as the name of the dat file, and also assumes that you will export `sample.dat` to the directory `C:\tibco\adapter\adcom\5.3\examples`. If you export it to a different directory, you must update the path in the `testinterceptor.dsw` and rebuild your COM client before you import the metadata into the project file.



TIBCO Adapter for COM ships with a local repository `sample.dat` in the `<Install_dir>\examples` directory. The `sample.dat` repository has predefined session and logging information, and two sample instances, one for an interceptor component (`TIBCOCOMInterceptor1`) and the other for a service component (`TIBCOCOMService1`). Therefore, it is recommended that you rename `sample.dat` to prevent it from being overwritten.

Permissions to Access Repository Server

If your site is using TIBCO Administrator to set access control to the repository server, you must have the account name and password available that is used by the adapter to log into the server.

Other Examples

A set of examples are included online in the `examples\TEAKEexamples` directory. For more information, see the *TIBCO Adapter for COM Examples Guide*. Additionally, a set of demos are provided in the `examples\Demos` directory. For more information, see *Demonstration Applications* on page 147. Also, for information on .NET examples, see *Interoperability with Microsoft .NET Components* on page 191.

Create the Project

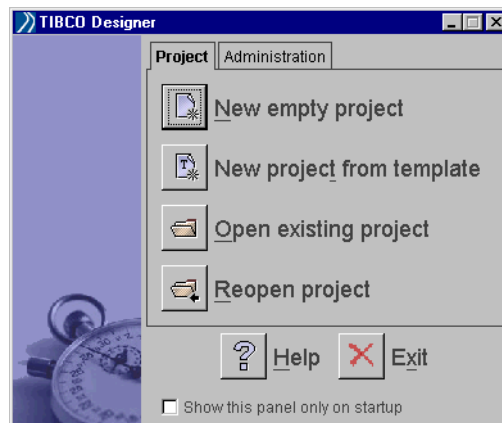
The TIBCO Designer GUI is used to configure adapter instances. When starting TIBCO Designer, you must create or select a project. A project contains the configuration files that define options used by a run-time adapter. After a project is configured, it is exported to a repository file and available for use by the run-time adapter.

To create a project:

1. Start TIBCO Designer by executing the following command, depending on your operating system.

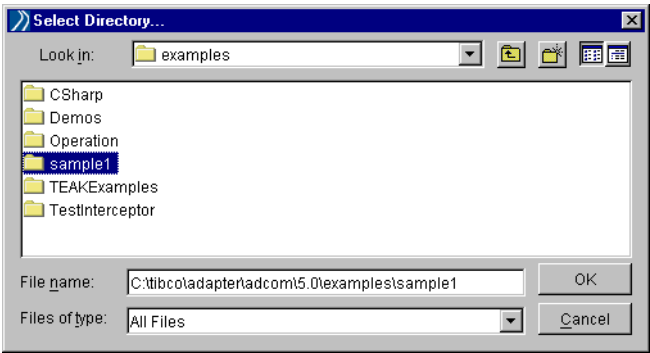
On Microsoft Windows, select: **Start>Programs>TIBCO>TIBCO Designer 5.2>Designer 5.2**

2. In the TIBCO Designer dialog, click **New empty project**.

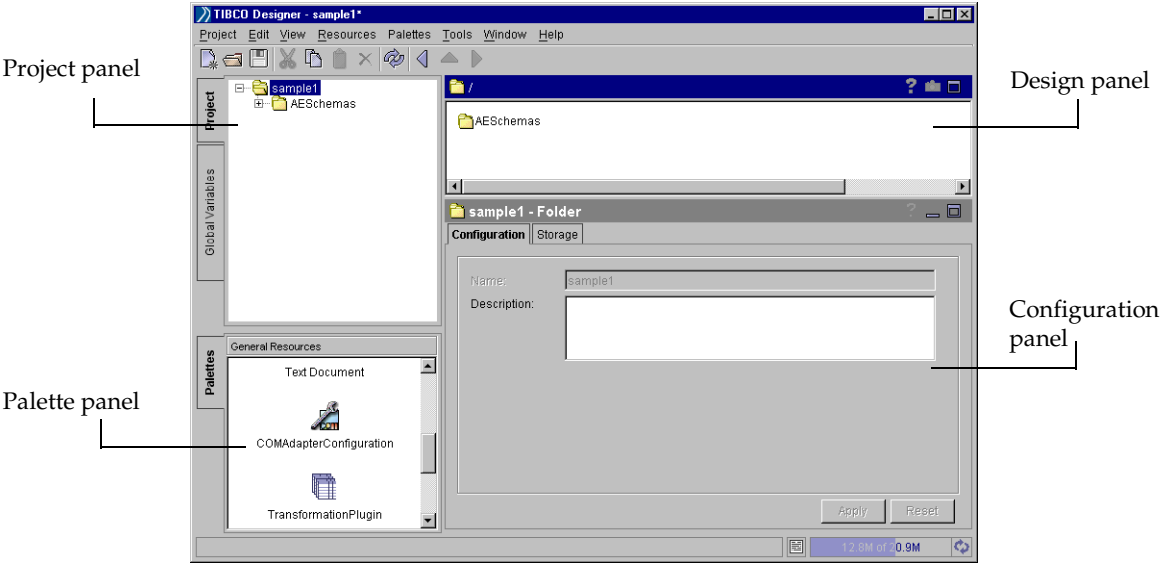


3. In the Save Project dialog, select **Multi-File Project** (if it is not selected) and click the **Browse** button. As shown in the next diagram, navigate to the C:\tibco\adapter\adcom\5.3\examples folder and click the Create New Folder icon. Name the new directory **sample1** and click **OK**. The files for the

example will be saved in the sample1 directory. In the Save Project dialog, click **OK**.



The next diagram shows the TIBCO Designer GUI with the sample1 project defined.



Create the COM Components

After you create your project and before you import metadata, you must build your COM server. You can also build your COM client at this time.

To create the COM server:

- Start Microsoft Visual C++ and open the `tibco\adapter\adcom\5.3\examples\Operation\Operation2\Operations2.dsw` workspace. Build `Operation1` and then `Operation2` by clicking **Build > Build <workspace>**.

To create the COM client:

- Start Microsoft Visual C++ and build the `TestInterceptor` project from the `tibco\adapter\adcom\5.3\examples\TestInterceptor` directory, using the `testinterceptor.dsw` workspace.



Both the `Operation1.dll` and `Operation2.dll` should be registered using the `regsvr32.exe` tool. To do so, run the following command from the command prompt:

```
regsvr32 <Operation1.dll_dir_path>/Operation1.dll
```

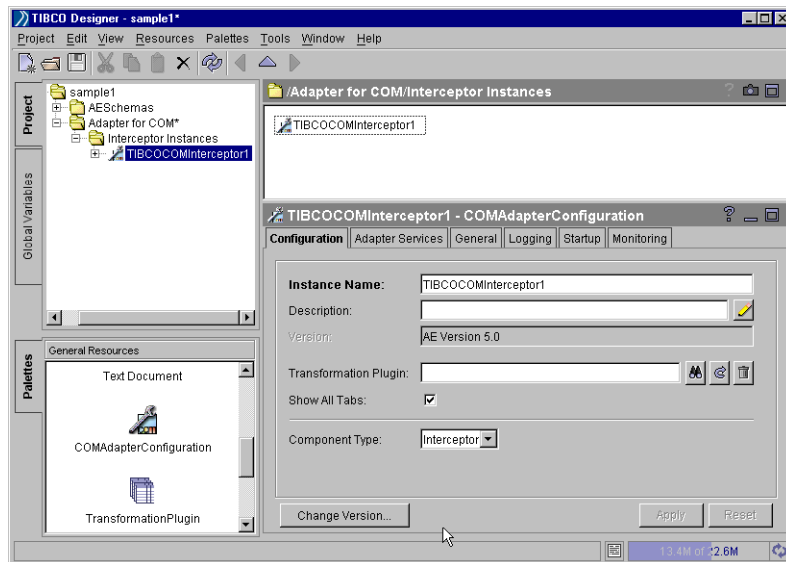
Configure the Adapter

Creating an adapter instance implies creating an interceptor or service instance, and then adding a service. An adapter instance can contain publication and request-response invocation services if it is an interceptor instance, and subscription and request-response services if it is a service instance. In this exercise, you will configure an interceptor instance with a publication service, and a service component with a subscription service.

Options for logging, startup and monitoring are set for an instance. In this exercise, default values are used for these options.

To configure the interceptor instance

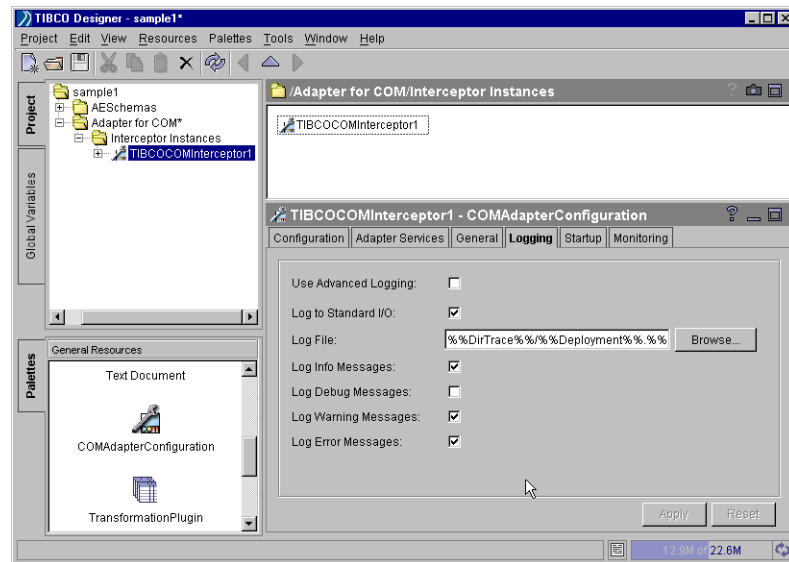
1. Drag and drop a **Folder** icon from the palette panel to the design panel. Rename it to **Adapter for COM**. Double-click the **Adapter for COM** folder on the tree in the project panel. Drag and drop another **Folder** icon from the palette panel to the design panel. Rename it to **Interceptor Instances**. Double-click the **Interceptor Instances** folder.
2. Drag the **COMAdapterConfiguration** icon from the palette panel to the design panel. This creates an adapter named by default, **COMAdapterConfiguration**. Change this name to **TIBCOCOMInterceptor1**.
3. Click **Apply**.



4. Click the **Logging** tab to identify the file log options.

In the next diagram, Information, Warning and Error messages are specified to be logged to the log file and standard input. The `Log File` field lists the global variables that are used to define the log file path and name. The `DirTrace` and `Deployment` variables are set using global variables. You can click the `Global Variable` tab to display the variables in the project panel. The default settings will be used in this example. The `InstanceId` variable need not be set. The variable automatically substitutes the adapter name at run-time.

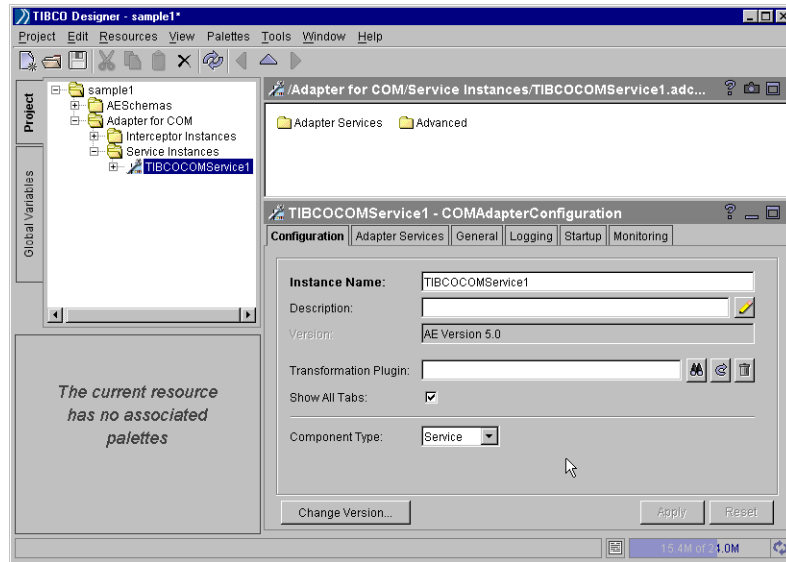
5. Click **Apply**.
6. Select **Project > Save**.



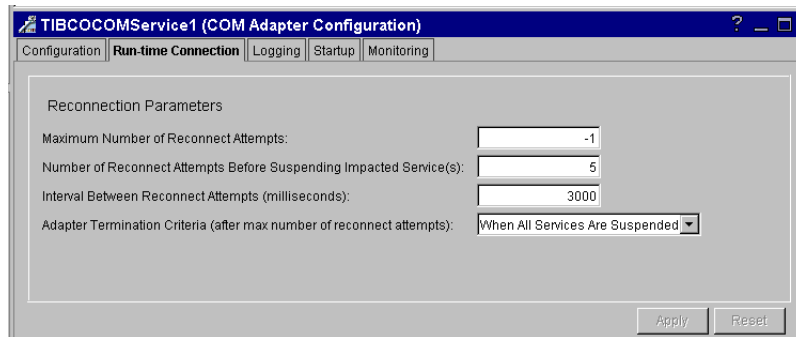
To configure the service instance

1. Double-click the `Adapter for COM` folder on the tree in the project panel. Drag and drop a `Folder` icon from the palette panel to the design panel. Rename it to `Service Instances`. Double-click the `Service Instances` folder.
2. Drag the `COMAdapterConfiguration` icon from the palette panel to the design panel. This creates an adapter named by default, `COMAdapterConfiguration1`. Change the name to `TIBCOCOMService1`.

3. Select **Service** in the **Component Type** drop-down. Click **Apply**.



4. Click the **Run-time Connection** tab to change any of the following default values. Click **Apply**.



5. Click the **Logging** tab to identify the file log options as specified in step 4 on page 40. Click **Apply**.
6. In the **Metadata Search URL** box under the **Startup** Tab, browse to the **AESchemas/ae/COM/OPERATION2Lib/UDT's** and select it.



Before you specify the URL, you must import the metadata as described in step 2 of the section **Configure the Publication Service** on page 43.

7. Click **Apply**.
8. Select **Project > Save**.

Configure the Publication Service

This section explains how to configure the interceptor instance with a publication service that will make a COM method call and send the message to TIBCO messaging transport.

Configure the Publication Service

1. In the project panel, expand the `TIBCOCOMInterceptor1` icon, then highlight the `Adapter Services` folder to access the `Publication Service` icon. Drag the icon from the palette panel to the design panel.
2. Click the `Adapter Services` folder in the project panel. Import the `operation1` and `operation2` type libraries into the `sample.dat` using the `Import Metadata` option on the `Import Schema` tab. The type libraries are available in the
`tibco\adapter\adcom\5.3\examples\Operation\Operation1` and
`tibco\adapter\adcom\5.3\examples\Operation\Operation2` directories.



You can import metadata into the service configured for either the `TIBCOCOMInterceptor1` or `TIBCOCOMService1` adapter instance.

3. Select `PublicationService` in the `Project` panel. In the `Configuration` tab:
 - a. Type `Operation2_Testobj2_1_Publisher` in the `Name` field.
 - b. Select `TIBCO Rendezvous` from the `Transport Type` drop-down.
 - c. Click **Apply**.
4. In the `Transport` tab:
 - a. Type `TIBCOCOM.Operation2.Testobj2.1` in the `Message Subject` field.
 - b. Select `Reliable` from the `Quality of Service` drop-down.
 - c. Retain the default values for the `Wire Format` drop-down.
 - d. Click **Apply**.
5. In the `SchemaView` tab, expand the `AESchemas` folder to specify a class reference. Drill down to the `AESchemas>ae>COM>OPERATION2Lib>coClasses` folder and select `Operation2.Testobj2.1`.
6. Click **Apply**.
7. Select **Project > Save**.

Configure the Subscription Service

This section explains how to configure a subscription service to get a message from the TIBCO messaging transport and invoke a COM method call on the COM server. The exercise assumes you have completed the previous exercise, Configure the Publication Service on page 43, and uses ActiveEnterprise classes created during the exercise. You must use the same project created in the previous exercise.

Configure the Subscription Service

1. In the project panel, expand the `TIBCOCOMService1` icon, then select the `Adapter Services` folder to access the `Subscription Service` icon. Drag the icon from the palette panel to the design panel.
2. In the Configuration tab:
 - a. Type `Operation2_Testobj2_1_Subscriber` in the `Name` field.
 - b. Select `TIBCO Rendezvous` from the `Transport Type` drop-down.
 - c. Click **Apply**.
3. In the Transport tab:
 - a. Type `TIBCOCOM_Operation2_Testobj2.1` in the `Message Subject` field.
 - b. Select `Reliable` from the `Quality of Service` drop-down.
 - c. Retain the default value for the `Wire Format` drop-down.
 - d. Click **Apply**.
4. In the SchemaView tab, expand the `AESchemas` folder to specify a class reference. Drill down to the `AESchemas>ae>COM>OPERATION2Lib>coClasses` folder, and select `Operation2_Testobj2.1`.
5. Click **Apply**.
6. Select **Project > Save**.

Convert the Project to a Repository File

Export the project to a local repository:

1. Select **Project>Export Full Project**.
2. In the Export Project dialog box, specify the project name and directory to save to. Click **OK**.



You must save the file as `sample.dat`.

TIBCO Adapter for COM ships with a local repository `sample.dat` in the `<Install_dir>\examples` directory. Therefore, it is recommended that you rename `sample.dat` to prevent it from being overwritten.

Deploy the Project and Start the Adapter

After configuring the adapter, you must create a properties file for the adapter.

Task A Deploy the Adapter

To create a properties file for the adapter:

1. Make a backup copy of `adcomservice.tra` for safety.
2. Change directory to the adapter bin directory:
`cd C:\TIBCO\adapter\adcom\5.3\bin`
3. Using a text editor, open the `adcomservice.tra` file and verify the following properties.
 Verify: `#TIBCO.repourl <repourl>` points to:
`TIBCO.repourl C:/tibco/adapter/adcom/5.3/examples/sample.dat`
 Verify: `#TIBCO.configurl <configurl>` points to:
`TIBCO.configurl Adapter for COM/Service Instances/TIBCOCOMService1`

Task B Start the Adapter

Open two command windows:

4. In one command window, go to `C:\tibco\adapter\adcom\5.3\bin` and start the `TIBCOCOMService1` service:
`adcomservice --run --propFile adcomservice.tra`
5. In the other command window, go to
`C:\tibco\adapter\adcom\5.3\Examples\TestInterceptor\Debug` and start the COM client:
`TestInterceptor`

The COM client invokes COM methods using the interceptor component. The publication service (`TIBCOCOMInterceptor1`) of the interceptor will publish the message to the TIBCO messaging transport. The subscription service (`TIBCOCOMService1`) of the service component retrieves the messages, and invokes the COM methods on the COM server.

Chapter 4

Adapter Instance Options

This chapter explains how to create an adapter instance and assign it services by configuring standard settings. All configuration tasks are performed in TIBCO Designer and the information is stored in a project that is later used by the run-time adapter.

Topics

- *Overview, page 48*
- *Adapter Instance Fields, page 49*
- *Adapter Services, page 58*
- *Publication Service Fields, page 61*
- *Subscription Service Fields, page 66*
- *Request-Response Service Fields, page 72*
- *Request-Response Invocation Service Fields, page 77*

Overview

In TIBCO Adapter for COM, configuring an adapter instance implies configuring an interceptor or a service component of the adapter.

Configuration Tasks

Use the following sequence to create and configure adapter services.

1. Start TIBCO Designer and open a multi-file project. See the *TIBCO Designer User's Guide* for details.
2. Drag the `COMAdapterConfiguration` icon from the palette panel to the design panel. This creates an adapter named, by default, `COMAdapterConfiguration`.
3. Define the adapter instance by assigning a new name and optionally change logging options. See *Logging Tab* on page 54 for details.
4. Specify whether you want to configure an interceptor or a service component of the adapter. To do so, in the *Configuration* tab, select one of the following from the *Component Type* drop-down:
 - *Interceptor*, to configure an interceptor component.
 - *Service*, to configure a service component.
5. Under the *Import Schema* tab, browse to select a type library and click **Import Metadata**. The *Import Schema* tab is available when you select the *Advanced Services* folder in the project panel.
6. Define the options under the *Adapter Services* tab and the *General* tab.
7. Add a service to the adapter instance by selecting the *Adapter Services* folder and dragging the service icon from the palette panel and dropping it in the design panel.
8. Define the configuration, transport, and schema options for the service.
Repeat step 2 through step 8 for each adapter service that you want to configure.
9. Export the project as a local repository and exit TIBCO Designer.

After configuring the adapter, you must create the run-time adapter property file and add the project name and adapter instance name. See *Set Properties in Properties File* on page 93 for details.

Adapter Instance Fields

The following tabs can be used to define an adapter instance:

- Configuration Tab on page 49
- Run-time Connection Tab on page 51
- Adapter Services Tab on page 52
- General Tab on page 53
- Logging Tab on page 54
- Startup Tab on page 55
- Monitoring Tab on page 56

Configuration Tab

Instance Name

Use the default name or replace it with a name of your choice.

- An instance name must use alphanumeric characters. An underscore (_) character can be used. The entire instance name must be less than 80 characters. The space character cannot be used in an instance name.
- An instance name cannot use global variables.
- An instance name must be unique with respect to other adapter instances for the same adapter in the project. The same instance name can be used to name an adapter instance for a different adapter in the same project. For example, an R/3 adapter instance named TEST and a Siebel adapter instance named TEST can coexist in the same project.
- Each instance name must be unique per adapter within a project even if each instance is defined in a different folder. That is, configuring same-named adapter instances in different folders will not make their names unique.

When you create an adapter instance, the palette automatically creates several resources for it. The names of these resources derive from the name of the instance they belong to. Changing the adapter instance name results in an automatic regeneration of the resources names. If you manually modify any resource name, that particular name will *not* be automatically regenerated next time you rename the adapter instance.

Description

Provide information about the adapter instance that you want stored in the project. The field is optional.

Version

The version string indicates the ActiveEnterprise (AE) format in which the adapter instance is saved. An adapter instance can be saved in AE Version 4.0 or AE Version 5.0, AE Version 5.1.0, AE Version 5.2, or AE Version 5.3 format.

When a new adapter instance is created in TIBCO Designer 5.x, the version string is set to AE Version 5.2. When a 4.x adapter instance is opened in TIBCO Designer 5.x, the Version field is set to AE Version 4.0.

- If a 4.x adapter instance is to be run against a 4.x run-time adapter, the instance must be saved with the Version field is set to AE Version 4.0.

If you are using TIBCO Designer 5.x to modify 4.x adapter instances, change only features supported by the 4.x. run-time adapter and use the validation utility to verify the instance before deploying the project. The validation utility scans the project and returns warnings if any 5.2 features are defined for 4.x adapter instances. Invoke the utility from the **Project>Validate Project for Deployment** menu command in TIBCO Designer.

- If a 4.x adapter instance is to be run against a 5.x run-time adapter, the Version field should be set to AE Version 5.2.

To change versions, click the **Change Version** button.

Message Filter

Specify a message filter, if you have configured a message filter resource for use with the adapter. The filter allows you to manipulate incoming and outgoing data before sending it on the network or handing it to the target application. Filters can be written using the TIBCO Adapter SDK. See the *TIBCO Adapter SDK Programmer's Guide* for information about writing a message filter.

Show All Tabs

Select this check box to display additional tabs for configuring advanced options.

Component Type

Select the run-time component you want to configure:

- **Interceptor** — This is selected by default. Use this to configure the interceptor component.

- **Service** — Select this to configure the service component.

Run-time Connection Tab

This tab is available when you select **Service** from the **Component Type** drop-down in the **Configuration** tab.

Maximum Number of Reconnect Attempts

Specify the number of reconnection attempts to make before the run-time adapter is stopped. A value of -1 implies that reconnection attempts will continue indefinitely. Zero is not a valid value.

Number of Reconnect Attempts Before Suspending Impacted Service(s)

Specify the number of reconnection attempts to make before suspending the adapter services.

Interval between Reconnect Attempts (milliseconds)

Specify the time interval in milliseconds to elapse between each reconnection attempt.

Adapter Termination Criteria (after max number of reconnect attempts)

The adapter provides the following choices:

- **When All Services Are Suspended** — To stop the adapter when a service is suspended. Therefore, only the adapter service that cannot reconnect is stopped. Other adapter services that are connected continue to function normally.
- **When Any Service is Suspended** — To stop the adapter if any one service is unable to re-establish a connection after the specified number of reconnection attempts.

Adapter Services Tab

Validate Metadata on Startup

This field is applicable only for the Publication and Request-Response Invocation services. By default, this check box is selected. When selected, the COM UDTs are validated to verify that they recursively match the TIBCO ActiveEnterprise classes they correspond to. This can be time-consuming.

When this check box is cleared, it turns off validation in the interceptor, which speeds up initialization of the interceptor.

In a testbed environment, you should select this option so that any inconsistencies between COM UDTs and TIBCO ActiveEnterprise classes or between COM interface methods and TIBCO ActiveEnterprise class operations will be flagged as errors. Such inconsistencies can occur if you make a change to your COM IDL file and rebuild your type library, but forget to make the corresponding change in (or reimport the type library as a whole into) the project file.

Once you have verified that the information in the COM type library and repository is consistent and you are moving into a production environment, clear this check box to allow the interceptor to initialize as quickly as possible.

Note that the interceptor performs extensive metadata validation at startup time and this validation is extremely useful in making sure that your testbed environment is consistent. So, clear this check box only after you are absolutely sure that you have removed all inconsistencies from your environment. If inconsistencies exist, unexpected runtime exceptions will occur.

Method Invocation Timeout (milliseconds)

This field is applicable only for the Publication and Request-Response Invocation services.

The default value is 60000. If the `interceptor` component of the adapter is mapping a COM method invocation to a two-way TIBCO Adapter SDK operation, the value of this parameter specifies how long the interceptor should wait for a reply. If the interceptor times out, it will return an error to the COM client.

Number of Threads

This field is applicable only for the Subscription and Request-Response services.

The default value is one. The value of this parameter specifies how many threads the service component of the adapter should use to process operation requests or published documents. Increasing this value will increase service throughput.

When a thread gets to the point of invoking the COM server method, the threading model of the COM server determines the behavior. If the threading model of the COM server is single or apartment, the methods are called sequentially. If the threading model of the COM server is free, or both, the methods are called in parallel.

General Tab

Termination Subject or Topic

A message sent on the termination subject (if TIBCO Rendezvous is the transport) or topic (if JMS is the transport) stops the adapter. In most cases, you should use the default value.

See *TIBCO Rendezvous Concepts* for information about specifying subject names. See the *TIBCO Enterprise for JMS User's Guide* for information about publishing on a topic.

Log Tracking Information

This field is available only for 4.x adapter configurations. For 5.0 adapter configurations, this field is preselected and unavailable.

Select this check box if you want to log adapter-specific trace information to files. Only errors generated by the adapter are logged to a trace file. Errors from other sources, such as TIBCO Adapter SDK APIs are printed to the console window where the adapter started.

The TIBCO Adapter SDK API (`MTrackingInfo`) provides support for end-to-end traceability of documents or messages throughout the system. The information allows you to track a message back to its source when an error occurs in a given component, and provides status information about the progress of a business process that is distributed among multiple components. For information on the different components of a trace message, see Appendix D, Trace Messages on page 231.

For error messages (`errorrole`), the complete tracking information is traced. For debug, informational, and warning messages, only the tracking ID (`trackingId`) is traced. Informational (`inforole`) messages are only recorded in the sink files specified in the project file, and not in the Event log or console.

Log to Microsoft Windows Event Log

Select this check box to enable logging of informational and error messages to the Microsoft Windows Event Log. Turning on Event Log logging also provides more extensive error logging than can be obtained simply by defining the different kinds of monitoring sinks described for the Logging Tab on page 54.



The interceptor, when run as a service, will use the updated value only after you restart the service.

Logging Tab

Use Advanced Logging

When `Use Advanced Logging` is not selected (the default), you can set two standard output destinations (sinks) for trace messages and set the tracing level for the roles selected.

When `Use Advanced Logging` is selected, you have complete control on selecting the destinations and associating roles with each of the destinations.

To create and configure the sinks, select the log sinks folder under the Advanced folder in the project panel.

To create sinks, drag and drop the `Generic log sink` icon from the palette panel into the design panel. From the configuration panel, select the sink type. The following are the sink types available:

- File
- Hawk
- Network
- STDIO

When File and STDIO sinks are created from the `Generic log sink` they offer further configuration options. For the File sink, the file limit, file count, and the option to append or overwrite can be specified. When created by default, this is set to 30000 bytes, 3 and append mode respectively. For the STDIO sink, the option to write to `stdout` or `stderr` can be selected. When created by default, `stdout` is selected.

The Hawk sink uses the hawk session, created and used by the adapter for monitoring purposes, to send tracing messages to the TIBCO Hawk monitor or Display. For details on TIBCO Hawk sessions, see [Using Global Variables](#) on page 118. The configuration for the Hawk sink involves specifying the `MicroAgent Name` that must be specified in the configuration panel.

The Network sink is used to publish tracing message on TIBCO Rendezvous. The configuration for the network sink involves specifying the session, and the subject on which the trace messages needs to be published.

For all the sinks, optionally the name and description for the sink can be provided.

Log to Standard I/O

(STDIO Sink) When selected, trace messages are displayed in the command prompt window where the adapter is started. When not selected, trace messages do not display in the window.

Log File

Specify the name of the log file (log sink) to which trace messages are written. Global variables can be used to specify the location of the log file. See Using Global Variables on page 118 for more information.

The roles available are Info, Debug, Warning, and Error messages. The trace message generated depends on the roles selected. Turning on the roles can affect the performance of the adapter. Therefore, it is recommended that you turn on the required roles only.

Log Info/Debug/Warning/Error Messages

Trace messages of the selected level(s) will be collected in the named log sink. You can configure what levels of trace messages you want logged, and where trace messages are sent. There are three types of logs (log sinks) that you can configure to hold trace messages, corresponding to three levels (roles) of trace messages, Information, Warning and Error. A fourth level of trace messages, Debug, is reserved and should not be enabled unless requested by the TIBCO Product Support Group. This option writes a lot of information to the log file and significantly reduces the speed of the adapter.

Startup Tab

Show Startup Banner

This field is selected by default. The startup banner displays the run-time adapter version, the infrastructure version on which the adapter is built, and copyright information in the console window when the adapter is started.

Metadata Search URL

If the adapter does not find the class description for a class, it will try to resolve it by referring to the schema location specified by this URL. The field specifies the location where the adapter searches for base schemas. The adapter searches for any schema that has been defined and saved at this location, and that should be loaded at startup.

While setting schema location for an adapter instance, the dependency of the type libraries needs to be considered. For example, the parameter of a method might be defined as TIBCO ActiveEnterprise type `any`, which corresponds to the COM type `VARIANT`. At run time, the type of this parameter may be a specific TIBCO ActiveEnterprise class associated with a specific COM UDT. Therefore, the adapter requires the location in which it should search for the class definition.

Select `Browse Resources` to specify the location in which the adapter should search for the metadata.

Monitoring Tab

Many of the following fields make use of global variables. Click the `Global Variables` tab in the project panel to enter a value for a global variable.

Enable Standard Microagent

Allows you to turn on or off the standard TIBCO Hawk Microagent. The way to turn it on or off is also configurable. By clicking the `globe` icon, a standard check box or text value (true or false) can be used to turn the standard microagent on or off.

Standard Microagent Name

This is the name for the standard microagent that will be registered with the TIBCO Hawk system. In most cases the default value is used. The `InstanceId` variable need not be set because it is automatically set at run time by the run-time adapter.

Enable Class Microagent

This field is predefined and cannot be changed. It specifies the instance or class specific standard TIBCO Hawk Microagent.

Class Microagent Name

This is the name for the class microagent that will be registered with the TIBCO Hawk system. In most cases the default value is used.

Default Microagent Session

This field is predefined and cannot be changed. It specifies the name of the TIBCO Rendezvous session that will be used by the standard, class, and custom microagents.

The session name and the corresponding session is automatically generated by TIBCO Designer. Do not change the session name or the session. However, you can modify the session parameters if required. Navigate to the `Sessions` folder under the `Advanced` folder to modify the session parameters.

Make sure you have set the correct parameter value for the global variables that correspond to the TIBCO Hawk configuration. If the session parameters are not set properly, the microagents will not display in the TIBCO Hawk Display.

Adapter Services

After configuring an interceptor or service component of the adapter, select one or multiple adapter services for the instance. The following sections describe the fields that are available to the adapter.

- Publication Service Fields on page 61
- Subscription Service Fields on page 66
- Request-Response Service Fields on page 72
- Request-Response Invocation Service Fields on page 77

Additionally, after you configure adapter services, you must also import the metadata into the project file, either for the interceptor or the service instance.

- Import Schema Tab on page 58.

Import Schema Tab

This tab is available if you select the `Adapter Services` folder in the project panel.

Select Metadata Type

You must select the type of metadata that you want to import:

- `COM Type Library`

Select this if you are importing metadata from an executable, a dynamic linked library, or a COM type library file.

- `.NET Assembly`

Select this if you are importing metadata from a .NET assembly. For details on how the adapter interoperates with Microsoft .NET, see Appendix B, *Interoperability with Microsoft .NET Components*, on page 191.

COM Type Library

Enter the location of the COM type library (`.tlb`), dynamic linked library (`.dll`), or executable (`.exe`) that contains the metadata, or click **Browse** to navigate to it.

.NET Assembly

Enter the location of the .NET assembly that contains the metadata, or click **Browse** to navigate to it.

For details on how the adapter interoperates with Microsoft .NET, see Appendix B, Interoperability with Microsoft .NET Components, on page 191.

Path to .NET Framework Installation Directory

Enter the path for the .NET Framework installation, or click **Browse** to navigate to it.

For details on how the adapter interoperates with Microsoft .NET, see Appendix B, Interoperability with Microsoft .NET Components, on page 191.

Import Metadata

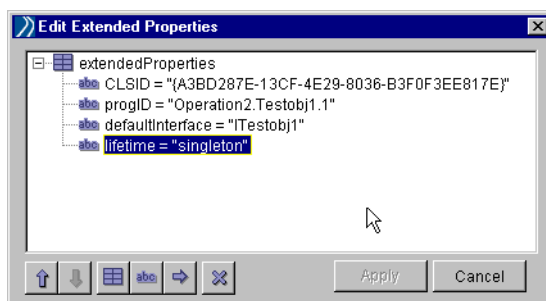
Click **Import Metadata** to import the specified metadata into the project file.

In a typical development environment, if rebuilding the COM server keeps changing the GUIDs, you must reimport the metadata into the project. Also, if the CLSIDs or IIDs are used explicitly, make the required changes. For example, use the Visual Basic development environment without turning on **Binary Compatibility** for the project.

Extended Properties

After you have imported the metadata, you may want to modify the extended properties for a class.

Select a class and click **Resources>Edit Extended Properties**. This displays the Edit Extended Properties dialog box, as shown below.



This dialog box contains a tree with the following entries:

- CLSID
- progID
- defaultInterface
- lifetime

The `lifetime` extended property can have two values. `singleton`(default), or `singleCall`. If the value is set to `singleton`, the instance is not released from memory after the first invocation. If the value is set to `singleCall`, the instance is released after each method call and a new instance is created when a method invocation request comes for the same object.

These four extended entries define the COM coclass/default interface pair that corresponds to the TIBCO ActiveEnterprise class associated with the adapter service.



When manually creating a TIBCO ActiveEnterprise class, you must add these extended properties. The property names are case sensitive.

Publication Service Fields

The following tabs are available:

- Configuration Tab on page 61
- Transport Tab on page 62
- SchemaView Tab on page 65
- Schema Tab on page 65

Configuration Tab

Name

You can use the default name or replace it with a name of your choice.

- A service name must use alphanumeric characters. An underscore (_) character can be used. The entire instance name must be less than 80 characters. The space character cannot be used in an instance name.
- A service name cannot use global variables.

Description

Provide information about the adapter service that you want stored in the project. The field is optional.

Transport Type

Select the transport to be used by the run-time adapter, JMS or TIBCO Rendezvous. After selecting the transport, the transport-specific configuration fields display.

The transport can be configured to use a trusted store and identity resource for use in SSL (Secure Sockets Layer) configurations. TIBCO Rendezvous sessions and JMS topics have an SSL configuration field which uses a dialog to perform SSL configuration.

To enable and configure SSL, in the **Project** panel, expand the **Advanced** folder, then expand the **Sessions** folder. Select the TIBCO Rendezvous session or JMS topic and click **Use SSL?**. The SSL configuration options are explained in the online help associated with the session dialog. Click the question mark to display the online help.

Transport Tab

Message Subject

This field displays only if `TIBCO Rendezvous` is selected in the `Transport Type` field (under the `Configuration` tab).

By default a service uses a message subject that is generated using the `Domain` and `Deployment` global variables, the adapter acronym, the adapter instance name and the service name. If you use this default subject, make sure the values for `Domain` and `Deployment` are not empty. You can type a `TIBCO Rendezvous` subject name different from the default in this field. See *TIBCO Rendezvous Concepts* for information about specifying subject names.

Destination

This field displays only if `JMS` is selected in the `Transport Type` field (under the `Configuration` tab).

By default a service uses a dynamic destination that is generated using the `Domain` and `Deployment` global variables, the adapter acronym, the adapter instance name and the service name. If you use this default dynamic destination, make sure the values for `Domain` and `Deployment` are not empty. You can override the default dynamic destination by specifying the static destination in this field. The static destination must be defined on the JMS server before it can be used by the run-time adapter. See the *TIBCO Enterprise for JMS User's Guide* for information about destinations.

Reply Message Subject

This field is not applicable to `TIBCO Adapter for COM`.

Reply Destination

This field is not applicable to `TIBCO Adapter for COM`.

Quality of Service

If `TIBCO Rendezvous` is selected as the transport type, select:

- `Reliable`

Ensures that each multicast or broadcast message is received as long as the physical network and packet recipients are working, and that the loss of a message is detected. This choice can compensate for brief network failures because it can retransmit a message on request if the first attempt failed. This

choice is appropriate when message delivery is expected but some loss can be tolerated.

- **Certified**

Guarantees that every certified message reaches its intended recipient in the order sent. The message can be sent across network boundaries, and if a network fails, delivery attempts continue until delivery succeeds or until the message's time limit expires. This is often called certified message delivery.

If certified message delivery is used, data is stored in a ledger file. The size of the ledger depends on several factors, the most important of which is the retention rate of stored data. That is, the ledger grows fastest in response to the cumulative length of undeliverable messages. You must ensure that sufficient disk space is available for the expected size of the ledger.

Wire Format

Services must use the same wire format to exchange data.

- **TIBCO Rendezvous Message** (TIBCO Rendezvous only)

Control information for validation is *not* sent in the message. If you use this format, the adapter is compatible with adapters not developed with TIBCO Adapter SDK.

- **ActiveEnterprise Message** (TIBCO Rendezvous only)

Control information for validation is sent in the message. If no control information is included, an exception is returned to the subscriber.

ActiveEnterprise standard wire format provides class information and packing rules for the TIBCO Adapter SDK set of data types. This format allows ActiveEnterprise components to perform extra validation on messages sent or received.

See the *TIBCO Adapter SDK Programmer's Guide* for details about the control information generated and sent with ActiveEnterprise messages.

- **XML Message** (TIBCO Rendezvous or JMS)

The XML Message wire format conforms to specifically constructed and fully compliant XML Schema (XSD) based on the existing definition of the ActiveEnterprise schema.

Connection Factory Type

- **Queue** (JMS only)

A message sent to a queue is consumed by one and only one receiver. Each message has only one receiver though multiple receivers may connect to the

queue. The first receiver to access the queue gets the message. The other receivers do not. This messaging model is known as point-to-point.

- **Topic (JMS only)**

A message published to a topic is broadcast to one or more subscribers. All messages published to the topic are received by all services that have subscribed to the topic. This messaging model is known as publish-subscribe.

Delivery Mode

- **Non Persistent (JMS only)**

A message marked non persistent will not be available to a JMS client if the JMS server goes down.

- **Persistent (JMS only)**

In general, a message marked persistent will be available to a JMS client even if the JMS server goes down.

Messages sent with the persistent delivery mode are always written to persistent storage, except when they are published to a topic that has no durable subscribers. When a topic has no durable subscribers, there are no subscribers that need messages resent in the event of a server failure. Therefore, messages do not need to be saved, and performance is improved because disk I/O is not required.

The semantics for these fields are somewhat more complex than the explanation given here. See the *TIBCO Enterprise for JMS User's Guide* for more information.

Session Reference

Every adapter instance can have one or more sessions configured for it. Sessions encapsulate stateful connections to TIBCO Rendezvous and other messaging sources. The session object shown in this field is initially supplied by the adapter, depending on the Quality of Service selected. You can change the session by browsing for it in the project panel.

Endpoint Reference

You can go to the referenced endpoint to edit its properties. Endpoint reference objects are explained in the TIBCO Designer *Palette Reference*.

SchemaView Tab

Schema Selection View

This view displays all the operation classes available under the AESchemas folder. Select an operation class from the list to attach a schema to the service.

Attempting to validate the configuration by clicking **Project > Validate for Deployment** before attaching a schema to the service may result in an error.

Once the schema has been specified for a service, you can view its attributes by selecting the AESchemas folder in the project panel. You can expand the field names to view the data types and field attributes. For more information about schema, see the *TIBCO Designer User's Guide*.

Schema Tab

Class Reference

After you have selected a schema in the SchemaView tab, this field is populated. This field is predefined and cannot be changed.

Subscription Service Fields

The following tabs are available:

- Configuration Tab on page 66
- Transport Tab on page 67
- SchemaView Tab on page 71
- Schema Tab on page 71

Configuration Tab

Name

You can use the default name or replace it with a name of your choice.

- A service name must use alphanumeric characters. An underscore (_) character can be used. The entire instance name must be less than 80 characters. The space character cannot be used in an instance name.
- A service name cannot use global variables.

Description

Provide information about the adapter service that you want stored in the project. The field is optional.

Transport Type

Select the transport to be used by the run-time adapter, JMS or TIBCO Rendezvous. After selecting the transport, the transport-specific configuration fields display.

The transport can be configured to use a trusted store and identity resource for use in SSL (Secure Sockets Layer) configurations. TIBCO Rendezvous sessions and JMS topics have an SSL configuration field which uses a dialog to perform SSL configuration.

To enable and configure SSL, in the **Project** panel, expand the **Advanced** folder, then expand the **Sessions** folder. Select the TIBCO Rendezvous session or JMS topic and click **Use SSL?**. The SSL configuration options are explained in the online help associated with the session dialog. Click the question mark to display the online help.

Message Confirm Behavior

You can control how the service component of the adapter confirms RVCN, by setting the message confirm behavior. A particular behavior can be selected by setting the `messageConfirmBehavior` extended property on an RVCN subscriber, when you are configuring the service instance in the project file.

This property is not available for RV subscribers (since RV messages do not require confirmation) or RV/RVCN RPC Servers (since certified messaging is typically not used in RPC).

- **Confirm Always**

This is the default behavior. The service component of the adapter always confirms the RVCN message, without regard to whether it can instantiate the instance of the COM coclass or what the `HRESULT` returned by the method call is.

- **Confirm on Instantiation**

If the service component of the adapter cannot instantiate the instance of the COM coclass, it does not confirm the message. Otherwise, it confirms the message, regardless of the `HRESULT` returned by the method call.

- **Confirm on Success**

If the service component of the adapter cannot instantiate the instance of the COM coclass or receives an unsuccessful `HRESULT` back from the method call, it does not confirm the message. Otherwise, it confirms the message.



The service component always logs an error message if it cannot instantiate the object or if it receives a unsuccessful `HRESULT` from the COM object. Therefore, the logging of an error message is independent of whether the message is confirmed. The error message usually does not contain the data from the message, but only an indication of what the error is. However, if the COM object has included such data in the error message, then the error message retrieved by the service component from the COM object using the `IErrorInfo` mechanism will contain the data.

Transport Tab

Message Subject

This field displays only if TIBCO Rendezvous is selected in the `Transport Type` field (under the `Configuration` tab).

By default a service uses a message subject that is generated using the `Domain` and `Deployment` global variables, the adapter acronym, the adapter instance name and the service name. If you use this default subject, make sure the values for `Domain` and `Deployment` are not empty. You can type a TIBCO Rendezvous subject name different from the default in this field. See *TIBCO Rendezvous Concepts* for information about specifying subject names.

Destination

This field displays only if JMS is selected in the `Transport Type` field (under the `Configuration` tab).

By default a service uses a dynamic destination that is generated using the `Domain` and `Deployment` global variables, the adapter acronym, the adapter instance name and the service name. If you use this default dynamic destination, make sure the values for `Domain` and `Deployment` are not empty. You can override the default dynamic destination by specifying the static destination in this field. The static destination must be defined on the JMS server before it can be used by the run-time adapter. See the *TIBCO Enterprise for JMS User's Guide* for information about destinations.

Quality of Service

If TIBCO Rendezvous is selected as the transport type, select:

- `Reliable`

Ensures that each multicast or broadcast message is received as long as the physical network and packet recipients are working, and that the loss of a message is detected. This choice can compensate for brief network failures because it can retransmit a message on request if the first attempt failed. This choice is appropriate when message delivery is expected but some loss can be tolerated.

- `Certified`

Guarantees that every certified message reaches its intended recipient in the order sent. The message can be sent across network boundaries, and if a network fails, delivery attempts continue until delivery succeeds or until the message's time limit expires. This is often called certified message delivery.

If certified message delivery is used, data is stored in a ledger file. The size of the ledger depends on several factors, the most important of which is the retention rate of stored data. That is, the ledger grows fastest in response to the cumulative length of undeliverable messages. You must ensure that sufficient disk space is available for the expected size of the ledger.

- `Distributed Queue`

Distributed queue includes a group of cooperating transport objects, each in a separate process. Each transport object is called a member. To balance the transmission load among servers, the adapter can use distributed queues for *one-of-n* delivery of messages to a group of servers. Each member of a distributed queue listens for the same subject using the TIBCO Rendezvous Distributed Queue listener objects. Even though many members listen for each inbound message (or task), only one member processes the message. For details on distributed queues, see *TIBCO Rendezvous Concepts*.

Load balancing for the processing of TIBCO Rendezvous certified messages is supported by using distributed queuing. The messages from TIBCO Rendezvous are distributed equally among all instances that belong to the same group. This distributes the message load over several adapter instances. However, the order in which messages are sent to the application is not guaranteed.

Wire Format

Services must use the same wire format to exchange data.

- `TIBCO Rendezvous Message` (TIBCO Rendezvous only)

Control information for validation is *not* sent in the message. If you use this format, the adapter is compatible with adapters not developed with TIBCO Adapter SDK.

- `ActiveEnterprise Message` (TIBCO Rendezvous only)

Control information for validation is sent in the message. If no control information is included, an exception is returned to the subscriber. ActiveEnterprise standard wire format provides class information and packing rules for the TIBCO Adapter SDK set of data types. This format allows ActiveEnterprise components to perform extra validation on messages sent or received.

See the *TIBCO Adapter SDK Programmer's Guide* for details about the control information generated and sent with ActiveEnterprise messages.

- `XML Message` (TIBCO Rendezvous or JMS)

The XML Message wire format conforms to specifically constructed and fully compliant XML Schema (XSD) based on the existing definition of the ActiveEnterprise schema.

Connection Factory Type

- `Queue` (JMS only)

A message sent to a queue is consumed by one and only one receiver. Each message has only one receiver though multiple receivers may connect to the queue. The first receiver to access the queue gets the message. The other receivers do not. This messaging model is known as point-to-point.

- **Topic (JMS only)**

A message published to a topic is broadcast to one or more subscribers. All messages published to the topic are received by all services that have subscribed to the topic. This messaging model is known as publish-subscribe.

Delivery Mode

- **Non-durable (JMS only)**

If a subscription service is marked non-durable, it indicates that messages will not be resent on the configured topic or queue, if the JMS server goes down.

- **Durable (JMS only)**

If a subscription service is marked durable, it indicates that messages need to be resent on the configured topic or queue, if the JMS server goes down.

Messages sent with the persistent delivery mode are always written to persistent storage, except when they are published to a topic that has no durable subscribers. When a topic has no durable subscribers, there are no subscribers that need messages resent in the event of a server failure. Therefore, messages do not need to be saved, and performance is improved because disk I/O is not required.

The semantics for these fields are somewhat more complex than the explanation given here. See the *TIBCO Enterprise for JMS User's Guide* for more information.

Session Reference

Every adapter instance can have one or more sessions configured for it. Sessions encapsulate stateful connections to TIBCO Rendezvous and other messaging sources. The session object shown in this field is initially supplied by the adapter, depending on the Quality of Service selected. You can change the session by browsing for it in the project panel.

Endpoint Reference

You can go to the referenced endpoint to edit its properties. Endpoint reference objects are explained in the TIBCO Designer *Palette Reference*.

SchemaView Tab

Schema Selection View

This view displays all the operation classes available under the AESchemas folder. Select an operation class from the list to attach a schema to the service.

Attempting to validate the configuration by clicking **Project > Validate for Deployment** before attaching a schema to the service may result in an error.

Once the schema has been specified for a service, you can view its attributes by selecting the AESchemas folder in the project panel. You can expand the field names to view the data types and field attributes. For more information about schema, see the *TIBCO Designer User's Guide*.

Schema Tab

Class Reference

After you have selected a schema in the SchemaView tab, this field is populated. This field is predefined and cannot be changed.

Request-Response Service Fields

The following tabs are available:

- Configuration Tab on page 72
- Transport Tab on page 73
- SchemaView Tab on page 76
- Schema Tab on page 76

Configuration Tab

Name

You can use the default name or replace it with a name of your choice.

- A service name must use alphanumeric characters. An underscore (_) character can be used. The entire instance name must be less than 80 characters. The space character cannot be used in an instance name.
- A service name cannot use global variables.

Description

Provide information about the adapter service that you want stored in the project. The field is optional.

Transport Type

Select the transport to be used by the run-time adapter, JMS or TIBCO Rendezvous. After selecting the transport, the transport-specific configuration fields display.

The transport can be configured to use a trusted store and identity resource for use in SSL (Secure Sockets Layer) configurations. TIBCO Rendezvous sessions and JMS topics have an SSL configuration field which uses a dialog to perform SSL configuration.

To enable and configure SSL, in the **Project** panel, expand the **Advanced** folder, then expand the **Sessions** folder. Select the TIBCO Rendezvous session or JMS topic and click **Use SSL?**. The SSL configuration options are explained in the online help associated with the session dialog. Click the question mark to display the online help.

Transport Tab

Message Subject

This field displays only if `TIBCO Rendezvous` is selected in the `Transport Type` field (under the `Configuration` tab).

By default a service uses a message subject that is generated using the `Domain` and `Deployment` global variables, the adapter acronym, the adapter instance name and the service name. If you use this default subject, make sure the values for `Domain` and `Deployment` are not empty. You can type a `TIBCO Rendezvous` subject name different from the default in this field. See *TIBCO Rendezvous Concepts* for information about specifying subject names.

Destination

This field displays only if `JMS` is selected in the `Transport Type` field (under the `Configuration` tab).

By default a service uses a dynamic destination that is generated using the `Domain` and `Deployment` global variables, the adapter acronym, the adapter instance name and the service name. If you use this default dynamic destination, make sure the values for `Domain` and `Deployment` are not empty. You can override the default dynamic destination by specifying the static destination in this field. The static destination must be defined on the JMS server before it can be used by the run-time adapter. See the *TIBCO Enterprise for JMS User's Guide* for information about destinations.

Quality of Service

If `TIBCO Rendezvous` is selected as the transport type, select:

- **Reliable**
Ensures that each multicast or broadcast message is received as long as the physical network and packet recipients are working, and that the loss of a message is detected. This choice can compensate for brief network failures because it can retransmit a message on request if the first attempt failed. This choice is appropriate when message delivery is expected but some loss can be tolerated.
- **Certified**
Guarantees that every certified message reaches its intended recipient in the order sent. The message can be sent across network boundaries, and if a network fails, delivery attempts continue until delivery succeeds or until the message's time limit expires. This is often called certified message delivery.

If certified message delivery is used, data is stored in a ledger file. The size of the ledger depends on several factors, the most important of which is the retention rate of stored data. That is, the ledger grows fastest in response to the cumulative length of undeliverable messages. You must ensure that sufficient disk space is available for the expected size of the ledger.

- **Distributed Queue**

Distributed queue includes a group of cooperating transport objects, each in a separate process. Each transport object is called a member. To balance the transmission load among servers, the adapter can use distributed queues for *one-of-n* delivery of messages to a group of servers. Each member of a distributed queue listens for the same subject using the TIBCO Rendezvous Distributed Queue listener objects. Even though many members listen for each inbound message (or task), only one member processes the message. For details on distributed queues, see *TIBCO Rendezvous Concepts*.

Load balancing for the processing of TIBCO Rendezvous certified messages is supported by using distributed queuing. The messages from TIBCO Rendezvous are distributed equally among all instances that belong to the same group. This distributes the message load over several adapter instances. However, the order in which messages are sent to the application is not guaranteed.

Wire Format

Services must use the same wire format to exchange data.

- **XML Message (JMS only)**

The XML Message wire format conforms to specifically constructed and fully compliant XML Schema (XSD) based on the existing definition of the ActiveEnterprise schema.

- **ActiveEnterprise Message (TIBCO Rendezvous only)**

Control information for validation is sent in the message. If no control information is included, an exception is returned to the subscriber. ActiveEnterprise standard wire format provides class information and packing rules for the TIBCO Adapter SDK set of data types. This format allows ActiveEnterprise components to perform extra validation on messages sent or received.

See the *TIBCO Adapter SDK Programmer's Guide* for details about the control information generated and sent with ActiveEnterprise messages.

Connection Factory Type

- **Queue (JMS only)**

A message sent to a queue is consumed by one and only one receiver. Each message has only one receiver though multiple receivers may connect to the queue. The first receiver to access the queue gets the message. The other receivers do not. This messaging model is known as point-to-point.

- **Topic (JMS only)**

A message published to a topic is broadcast to one or more subscribers. All messages published to the topic are received by all services that have subscribed to the topic. This messaging model is known as publish-subscribe.

Delivery Mode

- **Non-durable (JMS only)**

If a subscription service is marked non-durable, it indicates that messages will not be resent on the configured topic or queue, if the JMS server goes down.

- **Durable (JMS only)**

If a subscription service is marked durable, it indicates that messages need to be resent on the configured topic or queue, if the JMS server goes down.

Messages sent with the persistent delivery mode are always written to persistent storage, except when they are published to a topic that has no durable subscribers. When a topic has no durable subscribers, there are no subscribers that need messages resent in the event of a server failure. Therefore, messages do not need to be saved, and performance is improved because disk I/O is not required.

The semantics for these fields are somewhat more complex than the explanation given here. See the *TIBCO Enterprise for JMS User's Guide* for more information.

Session Reference

Every adapter instance can have one or more sessions configured for it. Sessions encapsulate stateful connections to TIBCO Rendezvous and other messaging sources. The session object shown in this field is initially supplied by the adapter, depending on the Quality of Service selected. You can change the session by browsing for it in the project panel.

Endpoint Reference

You can go to the referenced endpoint to edit its properties. Endpoint reference objects are explained in the *TIBCO Designer Palette Reference*.

SchemaView Tab

Schema Selection View

This view displays all the operation classes available under the AESchemas folder. Select an operation class from the list to attach a schema to the service.

Attempting to validate the configuration by clicking **Project > Validate for Deployment** before attaching a schema to the service may result in an error.

Once the schema has been specified for a service, you can view its attributes by selecting the AESchemas folder in the project panel. You can expand the field names to view the data types and field attributes. For more information about schema, see the *TIBCO Designer User's Guide*.

Schema Tab

Class Reference

After you have selected a schema in the SchemaView tab, this field is populated. This field is predefined and cannot be changed.

Request-Response Invocation Service Fields

The following tabs are available:

- Configuration Tab on page 77
- Transport Tab on page 78
- SchemaView Tab on page 80
- Schema Tab on page 81

Configuration Tab

Name

You can use the default name or replace it with a name of your choice.

- A service name must use alphanumeric characters. An underscore (_) character can be used. The entire instance name must be less than 80 characters. The space character cannot be used in an instance name.
- A service name cannot use global variables.

Description

Provide information about the adapter service that you want stored in the project. The field is optional.

Transport Type

Select the transport to be used by the run-time adapter, JMS or TIBCO Rendezvous. After selecting the transport, the transport-specific configuration fields display.

The transport can be configured to use a trusted store and identity resource for use in SSL (Secure Sockets Layer) configurations. TIBCO Rendezvous sessions and JMS topics have an SSL configuration field which uses a dialog to perform SSL configuration.

To enable and configure SSL, in the **Project** panel, expand the **Advanced** folder, then expand the **Sessions** folder. Select the TIBCO Rendezvous session or JMS topic and click **Use SSL?**. The SSL configuration options are explained in the online help associated with the session dialog. Click the question mark to display the online help.

Transport Tab

Message Subject

This field displays only if `TIBCO Rendezvous` is selected in the `Transport Type` field (under the `Configuration` tab).

By default a service uses a message subject that is generated using the `Domain` and `Deployment` global variables, the adapter acronym, the adapter instance name and the service name. If you use this default subject, make sure the values for `Domain` and `Deployment` are not empty. You can type a `TIBCO Rendezvous` subject name different from the default in this field. See *TIBCO Rendezvous Concepts* for information about specifying subject names.

Destination

This field displays only if `JMS` is selected in the `Transport Type` field (under the `Configuration` tab).

By default a service uses a dynamic destination that is generated using the `Domain` and `Deployment` global variables, the adapter acronym, the adapter instance name and the service name. If you use this default dynamic destination, make sure the values for `Domain` and `Deployment` are not empty. You can override the default dynamic destination by specifying the static destination in this field. The static destination must be defined on the JMS server before it can be used by the run-time adapter. See the *TIBCO Enterprise for JMS User's Guide* for information about destinations.

Quality of Service

If `TIBCO Rendezvous` is selected as the transport type, select:

- **Reliable**
Ensures that each multicast or broadcast message is received as long as the physical network and packet recipients are working, and that the loss of a message is detected. This choice can compensate for brief network failures because it can retransmit a message on request if the first attempt failed. This choice is appropriate when message delivery is expected but some loss can be tolerated.
- **Certified**
Guarantees that every certified message reaches its intended recipient in the order sent. The message can be sent across network boundaries, and if a network fails, delivery attempts continue until delivery succeeds or until the message's time limit expires. This is often called certified message delivery.

If certified message delivery is used, data is stored in a ledger file. The size of the ledger depends on several factors, the most important of which is the retention rate of stored data. That is, the ledger grows fastest in response to the cumulative length of undeliverable messages. You must ensure that sufficient disk space is available for the expected size of the ledger.

Wire Format

Services must use the same wire format to exchange data.

- `XML Message` (JMS only)

The `XML Message` wire format conforms to specifically constructed and fully compliant XML Schema (XSD) based on the existing definition of the `ActiveEnterprise` schema.

- `ActiveEnterprise Message` (TIBCO Rendezvous only)

Control information for validation is sent in the message. If no control information is included, an exception is returned to the subscriber.

`ActiveEnterprise` standard wire format provides class information and packing rules for the TIBCO Adapter SDK set of data types. This format allows `ActiveEnterprise` components to perform extra validation on messages sent or received.

See the *TIBCO Adapter SDK Programmer's Guide* for details about the control information generated and sent with `ActiveEnterprise` messages.

Connection Factory Type

- `Queue` (JMS only)

A message sent to a queue is consumed by one and only one receiver. Each message has only one receiver though multiple receivers may connect to the queue. The first receiver to access the queue gets the message. The other receivers do not. This messaging model is known as point-to-point.

- `Topic` (JMS only)

A message published to a topic is broadcast to one or more subscribers. All messages published to the topic are received by all services that have subscribed to the topic. This messaging model is known as publish-subscribe.

Delivery Mode

- `Persistent` (JMS only)

In general, a message marked persistent will be available to a JMS client even if the JMS server goes down.

- **Non Persistent (JMS only)**

A message marked non persistent will not be available to a JMS client if the JMS server goes down.

Messages sent with the persistent delivery mode are always written to persistent storage, except when they are published to a topic that has no durable subscribers. When a topic has no durable subscribers, there are no subscribers that need messages resent in the event of a server failure. Therefore, messages do not need to be saved, and performance is improved because disk I/O is not required.

The semantics for these fields are somewhat more complex than the explanation given here. See the *TIBCO Enterprise for JMS User's Guide* for more information.

Session Reference

Every adapter instance can have one or more sessions configured for it. Sessions encapsulate stateful connections to TIBCO Rendezvous and other messaging sources. The session object shown in this field is initially supplied by the adapter, depending on the Quality of Service selected. You can change the session by browsing for it in the project panel.

Endpoint Reference

You can go to the referenced endpoint to edit its properties. Endpoint reference objects are explained in the TIBCO Designer *Palette Reference*.

SchemaView Tab

Schema Selection View

This view displays all the operation classes available under the AESchemas folder. Select an operation class from the list to attach a schema to the service. Attempting to validate the configuration by clicking **Project > Validate for Deployment** before attaching a schema to the service may result in an error.

Once the schema has been specified for a service, you can view its attributes by selecting the AESchemas folder in the project panel. You can expand the field names to view the data types and field attributes. For more information about schema, see the *TIBCO Designer User's Guide*.

Schema Tab

Class Reference

After you have selected a schema in the SchemaView tab, this field is populated. This field is predefined and cannot be changed.

Chapter 5

Deploying and Starting an Adapter Using TIBCO Administrator Enterprise Edition



This chapter provides an overview of deploying, starting, stopping, and monitoring adapter services using the TIBCO Administrator Enterprise Edition interface. See the TIBCO Administrator documentation set for details about using Enterprise Edition.

Topics

- *Overview, page 84*
- *Create an EAR File in TIBCO Designer, page 85*
- *Deploy the Project, page 86*
- *Start or Stop the Adapter, page 87*
- *Monitor the Adapter, page 88*

Overview

An adapter service can be deployed, started and managed using TIBCO Administrator Enterprise Edition or TIBCO Administrator Repository Edition.

- Enterprise Edition must be used if your adapter is part of a TIBCO BusinessWorks integration project.
- Repository Edition is used for adapters that do not participate in TIBCO BusinessWorks processes. If you are using Repository Edition, see Chapter 6, Deploying and Starting the Adapter Using TIBCO Administrator Repository Edition, on page 89 for details.

The following table summarizes the features available in each edition.

Table 5 TIBCO Administrator Features

Feature	Enterprise Edition	Repository Edition
tomcat web server	Yes	Yes
TIBCO Administration Server	Yes	Yes
TIBCO Administration GUI	Yes	Yes
User Management module	Yes	Yes
Resource Management module	Yes	No
Application Management module	Yes	No

The modules are optional. That is, Repository Edition can be installed with or without the User Management module. The User Management module is required to set security options for adapter projects.

Create an EAR File in TIBCO Designer

Generate an Enterprise Archive file (EAR) that contains information on what you wish to deploy. This could be one or more adapter services, one or more TIBCO BusinessWorks process engines, or both.



Building an archive creates the EAR file, which you can then deploy from TIBCO Administrator. If you make changes to the business processes or adapter services included in the archive, you need to rebuild the archive. Saving the project does not affect the archive.

In TIBCO Designer, follow these steps to create an EAR:

1. Configure the adapter services.
2. Drag and drop the `Enterprise Archive` resource from the palette panel to the design panel.
3. Select the `Enterprise Archive`. Drag and drop the `Process Archive` resource from the `Process` palette panel to the design panel. If there are any processes in your project, configure them using the `Browse Resources` button.
4. If there are any configured adapter services in your project, an `Adapter Archive` resource becomes available in the `Adapter Resources` palette panel. Drag the `Adapter Archive` into the design panel and specify information in the `Configuration` tab, then click **Apply**.
5. Go to the `Enterprise Archive` and click **Build Archive** to create the EAR file.

See Also

See the *TIBCO Designer User's Guide* for more information about this procedure. The guide is available from the `Designer Help` menu.

Deploy the Project

Before deploying a project, the machine on which the adapter is installed must be part of a TIBCO administration domain. After you have installed the TIBCO Administration Server, any machine on which you install TIBCO Runtime Agent (required by an adapter) is automatically added to the administration domain. The TIBCO software installed on the machine is then visible and accessible via the TIBCO Administrator GUI.

When you deploy a project, startup scripts and other information about the different components are sent to the machines to which the components were assigned. The project data store and TIBCO Administration Server are updated with the deployed components.

To deploy a project:

1. Start TIBCO Administrator, and import the EAR file into TIBCO Administrator Enterprise Edition.
2. Assign adapter archives to adapters installed in the administration domain and likewise assign process archives to process engines.
3. Specify startup options for each adapter service.
4. If desired, set up your deployment for fault tolerance by specifying adapter services to run on more than one machine. The secondary services will run in standby mode until they are needed.

See Also

See the *TIBCO Administrator User's Guide* for an introduction to the TIBCO administration domain and detailed information about the above steps.

See the *TIBCO Administrator Server Configuration Guide* for fault tolerance information.

Start or Stop the Adapter

The TIBCO Administrator *Application Management* module allows you to start, and stop deployed applications.

To start an adapter service from the module:

1. In the Administrator GUI left pane, expand **Application Management** > *Application-Name* > **Service Instances**.
2. In the *Service Instances* panel, select the check box next to the adapter service.
3. Click the **Start Selected** button.

The status changes from *Stopped* to *Starting up* to *Started*.

4. To stop the adapter service, select it, and then click the **Stop Selected** button.

See Also

See the *TIBCO Administrator User's Guide* for more information.

Monitor the Adapter

TIBCO Administrator offers a number of monitoring options.

- Specify alerts and TIBCO Hawk rulebases for each machine in the domain.
- Specify alerts and Hawk rulebases for each adapter service.
- View the log for each adapter service.

See Also

See the *TIBCO Administrator User's Guide* for information about configuring the above monitoring options.

Chapter 6

Deploying and Starting the Adapter Using TIBCO Administrator Repository Edition

This chapter explains how to deploy an adapter, then start and stop it on the command line. Repository Edition is used to manage adapter projects.

Topics

- *Overview, page 90*
- *Export the Project, page 91*
- *Create a Properties File, page 92*
- *Set Properties in Properties File, page 93*
- *Assign Security Options, page 94*
- *Start or Stop the Adapter, page 95*
- *Monitor the Adapter, page 99*
- *Adapter Properties File, page 100*
- *Server Repository URL, page 104*
- *Local Project URL, page 109*
- *Install the Adapter as a Service on Microsoft Windows, page 111*

Overview

If your adapter is not used in a TIBCO BusinessWorks process, you can use TIBCO Administrator Repository Edition to manage adapter projects.

If your adapter is used in a TIBCO BusinessWorks project, you must use TIBCO Administrator Enterprise Edition to deploy and manage your projects. See Chapter 5, Deploying and Starting an Adapter Using TIBCO Administrator Enterprise Edition, on page 83 for details.

See Overview on page 84 for a description of features available in both Enterprise Edition and Repository Edition.

Export the Project

In TIBCO Designer, after configuring, verifying, and testing your project, export the project to a server repository.

1. Click **Project>Export Full Project**.
2. Click the **Server Repository** tab in Export Project.
3. Provide values for the required fields and click **OK**.

See Also

See the *TIBCO Designer User's Guide* for details about the required fields. The guide is available from the TIBCO Designer Help menu.

Create a Properties File

Before starting the adapter you must create a properties file or edit the default properties file. Each run-time adapter instance must have a unique properties file. The easiest way to deploy an adapter is to copy the default `adcomService.tra` file to a unique name and edit the properties defined in the file.

To create a properties file:

1. Navigate to the adapter installation area:
`cd <install-path>\tibco\adapter\adcom\5.3\bin`
2. Copy the default `adcomService.tra` file to a unique name.
3. Edit the predefined properties in the copied file as explained in Set Properties in Properties File on page 93.

Set Properties in Properties File

The properties file contains a set of predefined properties. At a minimum, the following properties must be defined in the adapter property file. For a description of all predefined properties, see Adapter Properties File on page 100.

```
tibco.repourl tibcr://sample.dat
tibco.configurl adcom/COMAdapterConfiguration
application.args adcomService -system:propFile
                  C:/tibco/adapter/adcom/5.3/bin/adcomSamplecm.tra
application.start.dir
                  C:/tibco/adapter/adcom/5.3/bin
```

- The `tibco.repourl` property identifies the project in which the adapter configuration is defined.
- The `tibco.configurl` property identifies the adapter instance to start.
- The `application.args` property identifies the properties file to pass to the adapter.
- The `application.start.dir` property identifies the working directory for the adapter.

See Also

Adapter Properties File on page 100.

Assign Security Options

You must have installed TIBCO Administrator Repository Edition with the User Management module to assign security options. Setting security options is not required, but recommended.

The User Management module allows authorized users to specify security options for a TIBCO administration domain. Security has two components:

- Authentication. Specify users and passwords.
- Authorization. Give users access to individual components in the TIBCO administration domain.

By default, the administrative user for the administration domain has privileges to create users and assign privileges. That user can create additional users with the same full access privileges if appropriate.

Roles

TIBCO Administrator allows you to create a roles tree and assign users to one or more roles. You can then perform authorization on a per-role basis.

This capability is critical, for example, if a large number of users need read access to information about running projects but only a few users should be authorized to start and stop the project.

Authorization

TIBCO Administrator allows privileged users to authorize users for GUI Access or Data Access (or both).

- GUI access is given on a per-tab basis, that is, users are given access to components of the TIBCO Administrator GUI. For each tab in the TIBCO Administrator GUI, privileged users can specify which users are allowed read or write privileges.
- Data access is given on a per-data store basis. A repository data store is associated with each project. You can set various levels of access (read, write, variables) for data stores of each deployed project and for the administration domain data store.

See Also

See the *TIBCO Administrator User's Guide* for details about creating roles and assigning authorization.

Start or Stop the Adapter

The run-time adapter can be run as a console application. The adapter can also be installed and started as a Microsoft Windows service.

Start the Adapter

The following command line starts the adapter from a command window using the default `adcomService.tra` properties file. The default file has been edited with the name of the project, the adapter configuration and properties file to pass to the application. Because the properties file has the same name prefix as the executable and is in the same folder, it need not be specified on the command line.

```
adcomService --run
```

You can run the adapter as a console application using a custom properties file. For example, the next command line starts the adapter service that is identified in the `adcomSamplecm.tra` properties file, which is located in the same directory as the executable. The absolute pathname to the properties file must be given if it is located in a different directory than the executable.

```
adcomService --run --propFile adcomSamplecm.tra
```

Adapter Command Line Options

The next table describes the command line options for the run-time adapter.

Table 6 Adapter Command Line Options

Option	Description
--help	Displays help about the executable's command-line options.
--runAsNTService <service name>	Used only when the service component of the adapter is run as a Microsoft Windows service. This command line option is automatically added to the Image Path value when the service component is installed as a Microsoft Windows service.

Table 6 Adapter Command Line Options

Option	Description
<code>--createNTService <service name></code>	Adds the <code>adcomService</code> executable to the Microsoft Windows Services database with the specified service name. This option must be used in conjunction with the <code>-system:repourl</code> and <code>-system:configurl</code> options to specify the repository and configuration URLs that the Microsoft Windows service should use.
<code>--deleteNTService <service name></code>	Deletes the <code>adcomService</code> executable with the specified service name from the Microsoft Windows Services database. Make sure that the line, <code>application.args -system:propfile <file name></code> is commented out in the <code>adcomService.tra</code> file.
<code>--system:repourl <url></code>	Specifies which repository URL the service component should use.
<code>-system:configurl <url></code>	Specifies which adapter URL the service component should use.
<code>--system:clientVar <varName>=<value></code>	Allows you to define the value for a client variable in the repository. This value takes precedence over any global value set in the repository. Substitution takes place only at start up. Multiple <code>-system:clientVar</code> can be specified in the command line. If the same variable already exists in the command line, the latter overrides the former. No space is allowed for both the <code>varName</code> and the value when using <code>-system:clientVar</code> . For more details, see the <i>TIBCO Designer User's Guide</i> .
<code>--system:propfile <filepath></code>	Directs the adapter to load a command properties file containing startup information. These properties can be queried via the <code>MAppProperties::setProperty()</code> and <code>MAppProperties::getProperty()</code> methods (in C++ only).
<code>--run --propFile <fileName></code>	Launches the adapter using the specified properties file. This is the default option, if no options are specified on the command line.

Table 6 Adapter Command Line Options

Option	Description
<code>--install --propFile <fileName></code>	<p>Microsoft Windows Only. Installs the application as a Microsoft Windows Service using the specified properties file.</p> <p>If <code>propFile</code> is not given, the property file prefix name must have the same name as the adapter executable prefix name.</p>
<code>--uninstall --propFile <fileName></code>	<p>Microsoft Windows Only. Uninstalls the Microsoft Windows Service associated with the adapter.</p> <p>If <code>propFile</code> is not given, the property file prefix name must have the same name as the adapter executable prefix name.</p>
<code>--start --propFile <fileName></code>	<p>Microsoft Windows Only. Start the Microsoft Windows Service using the specified properties file.</p> <p>If <code>propFile</code> is not given, the property file prefix name must have the same name as the adapter executable prefix name.</p>
<code>--stop --propFile <fileName></code>	<p>Microsoft Windows Only. Stops the Microsoft Windows Service using the specified properties file.</p> <p>If <code>propFile</code> is not given, the property file prefix name must have the same name as the adapter executable prefix name.</p>



If the name and location of the properties file is different from the default properties file name and location, you must specify the absolute path at the command line.

The following examples illustrate typical uses of the command-line options for the service component:

Example 1 The following command line creates the Microsoft Windows service `TIBCOCOMService1`:

```
adcomService -createNTService TIBCOCOMService1 -system:repourl
"c:\tibco\adapter\adcom\5.3\examples\sample.dat" -system:configurl
"Adapter for COM/Service Instances/TIBCOCOMService1"
```

The service is created in the Microsoft Windows Services database, and is automatically started because the following command line is added when it is installed as a Microsoft Windows service:

```
adcomService -runAsNTService TIBCOCOMService1 -system:repourl
"c:\tibco\adapter\adcom\5.3\examples\sample.dat" -system:configurl
"Adapter for COM/Service Instances/TIBCOCOMService1"
```

Example 2 The following command line deletes the Microsoft Windows service TIBCOCOMService1 from the Microsoft Windows Services database:

```
adcomService -deleteNTService TIBCOCOMService1
```

Example 3 The following command line starts the adcomService executable as a standalone executable with the specified repository and adapter instances:

```
adcomService -system:repourl
"c:\tibco\adapter\adcom\5.3\examples\sample.dat" -system:configurl
"Adapter for COM/Service Instances/TIBCOCOMService1"
```

Stop the Adapter

Use one of the following methods to stop the adapter:

- From a command prompt window, close the window in which the adapter is running.
- From another command window, stop the adapter by sending a message on the terminate subject or terminate topic. See General Tab on page 53 for information about specifying the terminate subject or topic.
- From the **Control Panel > Services** dialog, stop the adapter service, if it is running as a Microsoft Windows service.

Monitor the Adapter

If you have installed TIBCO Hawk, you can use the methods defined in the adapter's standard and custom microagents to monitor it.

See Also

For details about monitoring, see *Appendix C, Monitoring the Adapter Using TIBCO Hawk*, on page 197.

Adapter Properties File

The run-time adapter parses a properties file at startup.

- The default run-time adapter properties file is named `adcomService.tra`.

The default properties file is located in `bin` subdirectory of the adapter installation directory.

Each line in a properties file is a single property. Each property consists of a key and a value. The key starts with the first non-whitespace character and ends at the first "=", ":", or whitespace character. The value starts at the first character after the equal sign (=). For example:

```
tibco.configurl=/tibco/private/adapter/test/config/config1
tibco.repourl=tibcr://TEST_PROJECT
tibco.username=admin
tibco.password=samplePassword
tibco.clientVar.service=7600
tibco.clientVar.daemon=tcp:7600
```

Properties defined in the properties file override the same properties defined in the project.

Properties File Format

The following restrictions apply to properties:

- The "!" character may not be used as a comment line indicator. Only the "#" character is recognized.
- The line continuation character is ignored (a value must fit on a line).
- The key may not contain any of the termination characters. Java allows termination characters by escaping the value with a preceding "\" character. The adapter does **not** support this syntax.

Predefined Properties

The next table describes predefined properties. Properties that start with `ntservice` are available only on Microsoft Windows platforms.



All paths inside a properties file, including Microsoft Windows directory names, must use forward slashes.

Table 7 Predefined Properties for Adapter Service

Property	Description
<code>tibco.repourl</code>	Identifies the absolute pathname to the TIBCO Designer project where the adapter configuration is defined. See Server Repository URL on page 104 for information about the locator string.
<code>tibco.configurl</code>	<p>Specifies the location of the adapter configuration inside the project file.</p> <ul style="list-style-type: none"> • If a relative path is specified, the adapter service is assumed to be under the default area in the project: <code>/tibco/private/adapter/</code> • If an absolute path is specified, the adapter configuration is looked up in the project as defined by the argument.
<code>tibco.username</code>	The user name and password used by the repository server to access the project.
<code>tibco.password</code>	
<code>adcom.perfMon <on/off></code>	Turns the performance statistics microagent on or off.
<code>adcom.addCustomHawkMethodstoStdMAgent <on/off></code>	If set to on, custom methods are added to the standard microagent. If set to off, custom methods are not available from the standard microagent. Default is on.
<code>adcom.dispcount</code>	Specifies the polling interval at which the adapter should poll the configured JMS queues, when JMS transport is used. The valid values are between 100 - 1000. To achieve stability, setting an optimum value, which is 500, is recommended. For example, <code>adcom.dispcount 500</code> .
<code>ntservice.name</code>	<p>Name for this Microsoft Windows Service.</p> <p>This property is useful if you wish to have multiple Microsoft Windows Services for the same executable. That is, you may wish to have two adapters running on the same machine. By specifying different service names and display names for the adapters, you can accomplish this.</p>

Table 7 Predefined Properties for Adapter Service

Property	Description
<code>ntservice.displayname</code>	<p>Name to display in the Services control for this Microsoft Windows Service.</p> <p>This property is useful if you wish to have multiple Microsoft Windows Services for the same executable. That is, you may wish to have two adapters running on the same machine. By specifying different service names and display names for the adapters, you can accomplish this.</p>
<code>ntservice.starttype</code>	<p>Start type for this Microsoft Windows Service. Either manual or automatic. For example: <code>ntservice.starttype=automatic</code></p> <p>You can use this property to initially set the start type for the service, but once the service is installed, use the Microsoft Windows Services control to change the start type of services.</p>
<code>ntservice.binary.path.absolute</code>	<p>Absolute path to the executable that is run when the service is started. For example: <code>ntservice.binary.path.absolute=C:/tibco/adapter/adcom/5.3/bin/adcomService.exe</code></p>
<code>ntservice.interactive</code>	<p>Specifies whether the Microsoft Windows Service is interactive. Either true or false. <code>ntservice.interactive=true</code></p>
<code>ntservice.account</code>	<p>Username under which to run the Microsoft Windows Service.</p> <p>You can use this property to initially set the account for the service, but once the service is installed, use the Services control to change the user account of services.</p>
<code>ntservice.password</code>	<p>Password for the username in the <code>ntservice.account</code> property.</p> <p>You can use this property to initially set the password for the user account, but once the service is installed, use the Services control to change the password.</p>

Table 8 Predefined Properties for Adapter Interceptor

Property	Description
<code>tibco.clientVar.<varname></code>	<p>Specifies run-time values to substitute for global variables defined in the project. This value takes precedence over the named global value set in the project. Substitution takes place only at start up.</p> <p>You append the global variable to <code>tibco.clientVar</code>, then give its value. For example: <code>tibco.clientVar.DirLedger=C:/tibco/adapter/adcom/5.3/myledger</code></p>
<code>adcom.AdStartTime</code>	<p>Specifies the start up timeout value of the adapter's Interceptor component, in minutes, when a repository file is used over the HTTP protocol. The valid values are between 5 - 50 minutes. For example, <code>adcom.AdStartTime 5</code>.</p>

Server Repository URL

The server repository URL identifies the project to load when the adapter starts. The string is given as a value to the `tibco.repourl` property which is specified in the adapter’s properties file.

The property syntax differs depending on the protocol with which the adapter and repository server communicate.

- TIBCO Rendezvous
- HTTP and HTTPS

TIBCO Rendezvous

In case of a TIBCO Rendezvous transport, the locator string begins with `tibcr://` or `tibcr@`. Next comes the project name.

In addition, the following optional properties are supported. They are separated by colons:

Table 9 Optional properties for server-based URL (TIBCO Rendezvous)

Property	Description
daemon	TIBCO Rendezvous rvd daemon value
service	TIBCO Rendezvous rvd service value
network	TIBCO Rendezvous rvd network value
rva	TIBCO Rendezvous rva host and port
subject	Instance discovery subject
discoveryTime	Timeout value in seconds for instance discovery
timeout	Timeout value in seconds for server requests
operationRetry	Number of retries when timeout occurs
userName	Any identifier (null or empty implies read only with guest privileges)
password	User password for security.

Table 9 Optional properties for server-based URL (TIBCO Rendezvous)

Property	Description
regionalSubject	TIBCO Rendezvous subject prefix used for regional read-operation in the load balancing mode. For additional information see the <i>TIBCO Administrator Server Configuration Guide</i> .
typeAccess	Type of adapter connection. Valid values are: <ul style="list-style-type: none"> CLIENT_USAGE_DONT_CARE Adapter reads until update, then switches to write. This is the default. CLIENT_USAGE_READ_ONLY Adapter is not allowed to do updates. CLIENT_USAGE_READ_WRITE Adapter can do both reads and updates.
urlFile	Path to the adapter's properties file. The property file identifier can either be a fully qualified path or a relative path. The legal properties in this file are the same as optional properties specified above. The properties in the file are appended to the repository locator string. If the same property appears in both locator string and property file, the properties in the locator string take precedence. Property values starting with # are considered obfuscated.

Examples

```
tibcr://myInst:service=5456:userName=ann:timeout=4000
tibcr@myInst:service=5456:urlFile=/tibco/props/fredsProps.txt
tibcr://myInst:urlFile=/tibco/props/fredsProps.txt
```

HTTP and HTTPS

In case of HTTP transports url begins with `http://`.

In case of HTTPS transports url begins with `https://`.

Host name and port number are next (`http://host:port`). The port number is optional. If it is not specified, the default value used is 8080 for HTTP and 8443 for HTTPS.

The host name and port number are followed by the project name, which is preceded by a question mark (?), for example, `http://host:8080/?inst1`

Optionally, `administrator/repo` may be included as part of the instance name, for example, `http://host:8080/administrator/repo/?inst1`



HTTPS-specific properties should be placed in a property file and that file should be specified using `"urlFile="`. `urlFile` is therefore a required property for HTTPS.

In addition, remote HTTP/HTTPS adapters support the following optional properties separated by `&`. Note that when `&` is used as the properties separator, and the URL is specified on the command line, the URL should be enclosed in quotes so that shell does not interpret it.

Table 10 *Optional properties for server-based URL (HTTP)*

Property	Description
timeout	Timeout value in seconds for server requests.
operationRetry	Number of retries if a timeout occurs.
userName	Any identifier (null or empty implies read only with guest privileges).
password	User password for security.
typeAccess	Whether it is read only or read-write. Valid values are: <ul style="list-style-type: none">CLIENT_USAGE_DONT_CARE Adapter reads until update, then switches to write. This is the default.CLIENT_USAGE_READ_ONLY Adapter is not allowed to do updates.CLIENT_USAGE_READ_WRITE Adapter can do both reads and updates.

Table 10 Optional properties for server-based URL (HTTP)

Property	Description
urlFile	<p>Path to the adapter's properties file.</p> <p>The property file identifier can either be a fully qualified path or a relative path. The legal properties in this file are the same as optional properties specified above. The properties in the file are appended to the repository locator string. If the same property appears in both locator string and property file, the properties in the locator string take precedence. Property values starting with # are considered obfuscated. See Defining a urlFile to be Accessed via HTTPS on page 107.</p>

Examples

```

http://host:8080?myInst&userName=ann&timeout=4000
http://host:8080/administrator/repo?myInst&userName=ann&timeout=4000
https://host:8443?myInst&urlFile=httpsProps.ini
https://host:8443/administrator/repo?myInst&urlFile=httpsProps.ini
https://host:8443/administrator/repo?urlFile=httpsProps.ini

```

Defining a urlFile to be Accessed via HTTPS

If the URL file is to be accessed via HTTPS, the following information must be included in the properties file:

Table 11 Properties Required for HTTPS urlFiles

Property	Description
trustedCertFormat	Format of the SSL certificate. Can be one of P12, PEM, DER (or ANS1).
httpsVendor	<p>Name of the SSL provider.</p> <p>The property is ignored because openssl is the only vendor supported.</p>
keyFile	<p>Key file.</p> <p>Keys can either be embedded as in the case of P12 and DER or non-embedded as in the case of PEM. Key file is relevant only in case of non-embedded key files, that is, PEM.</p>

Table 11 Properties Required for HTTPS urlFiles

Property	Description
identityFile	Location of the identity file.
identityType	Format of the identity file. Can be one of P12, PEM, DER (or ANS1).
trustedCertPassword	Password for the certificate specified by trustedCerts.
trustedCerts	Location of the trusted certificate or certificate chain.

Examples

Following is an example for a url file containing HTTPS specific properties:

```
httpsVendor=j2se
trustedCerts=H:/downloads/certs/clientcerts/trustedcerts/RSA/PEM/RSA1024ca1.cert.PEM
trustedCertFormat=PEM
trustedCertPassword=RSA1024ca1
identityFile=H:/downloads/certs/clientcerts/idcert/RSA/P12/RSA1024ca2.cert.P12

identityType=P12
keyPassword=RSA1024ca2
```

Local Project URL

Local repositories start with the instance name, which can optionally be preceded by localrepo@. The instance name can either be a fully qualified path or a relative path. The .dat extension is optional.

In addition, clients support the following optional parameters separated by colons:

Table 12 Optional parameters for local project URL

Parameter	Description
userName	Any identifier (if not present or empty makes a read-only client).
urlFile	<p>Path to the adapter's properties file.</p> <p>The property file identifier can either be a fully qualified path or a relative path. The legal properties in this file are the same as optional properties specified above. The properties in the file are appended to the repository locator string. If the same property appears in both locator string and property file, the properties in the locator string take precedence. Property values starting with # are considered obfuscated.</p>
typeAccess-	<p>Type of client connection. Valid values are:</p> <ul style="list-style-type: none"> CLIENT_USAGE_DONT_CARE Adapter reads until update, then switches to write. This is the default. CLIENT_USAGE_READ_ONLY Adapter is not allowed to do updates. CLIENT_USAGE_READ_WRITE Adapter can do both reads and updates. CLIENT_USAGE_REACQUIRE_INSTANCE_LOCK Adapter is allowed to overwrite this local repository even if a lock file exists, as long as it's the same user. CLIENT_USAGE_FORCE_INSTANCE_LOCK Adapter is allowed to overwrite this local repository even if a lock file exists. WARNING: Using this option may result in other users overwriting your project. <p>See the <i>TIBCO Administrator Server Configuration Guide</i> for additional information.</p>

Examples

```
myProj.dat  
myProj  
myProj/myrepo.dat
```

Install the Adapter as a Service on Microsoft Windows

The following command lines show how to install the adapter as a service, start the service, stop the service, then uninstall the service. Because no properties file is specified, the default properties file, <adapter_acronym>agent.tra is used.

1. Install the adapter as a service:

```
adcomService --install
```

2. Start the adapter service:

```
adcomService --start
```

3. Stop the adapter service:

```
adcomService --stop
```

4. Uninstall the adapter service:

```
adcomService --uninstall
```

If the properties file is not in the default location with the default name, you must explicitly specify the absolute pathname of the properties file on the command line.

You can create multiple services for the same adapter configuration by specifying a different Service Name and Display Name for each. This may be required for load balancing, if the subscriber quality of service is set to Distributed Queue. Each adapter configuration must have a unique properties file and it must be specified on the command line. In addition, each property file should have a unique value for the following Service Name and Display Name properties:

```
ntservice.name
ntservice.displayname
```

For example, the following command line lines show how to install the adapter as a service, start the service, stop the service, then uninstall the service using the adcomSample.tra property file. The same commands can be used to install another adapter on the same machine, provided a different properties file is given.

5. Install the adapter as a service:

```
adcomService --install --propFile adcomSample.tra
```

6. Start the adapter service:

```
adcomService --start --propFile adcomSample.tra
```

7. Stop the adapter service:

```
adcomService --stop --propFile adcomSample.tra
```

8. Uninstall the adapter service:

```
adcomService --uninstall --propFile adcomSample.tra
```


Chapter 7 **Advanced Topics**

This chapter explains advanced topics such as defining a TIBCO Hawk session, using global variables, setting encoding options for globalization, using DCOM and achieving high availability.

Topics

- *Overview, page 114*
- *Using the Adapter with a Revision Control System, page 115*
- *Configuring a TIBCO Hawk Session, page 116*
- *Using Global Variables, page 118*
- *Setting Encoding Options, page 122*
- *Using DCOM, page 123*
- *Using COM+, page 124*
- *Configuring Highly Available Instances, page 125*
- *Configuring the Adapter for Load Balancing Using RVCMQ, page 126*
- *Configuring the Adapter for Load Balancing Using JMS, page 128*
- *Exception Handling, page 129*
- *Obtaining Extended Error Information, page 131*

Overview

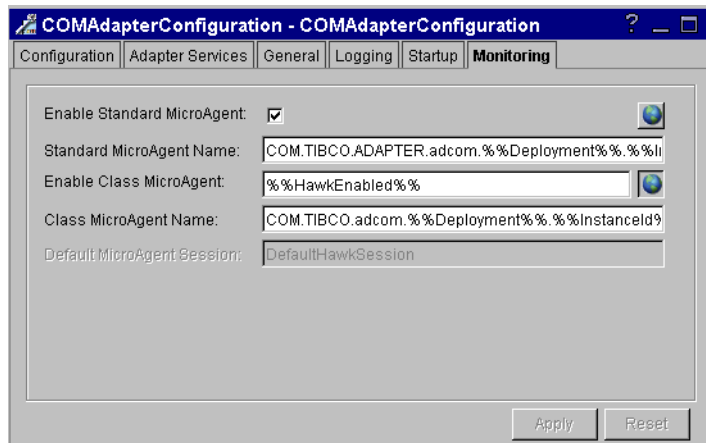
The advanced features of the adapter help you to perform additional functions. You can define a TIBCO Hawk session to monitor the adapter. You can also use variable substitution to override global variables that are predefined in the project. Additionally, you can configure the adapter to operate in a DCOM scenario or achieve high availability for the service component of the adapter.

Configuring a TIBCO Hawk Session

To use TIBCO Hawk to monitor the adapter you must first configure a TIBCO Rendezvous session on which TIBCO Hawk messages will be sent and received. Use the following steps to configure a session.

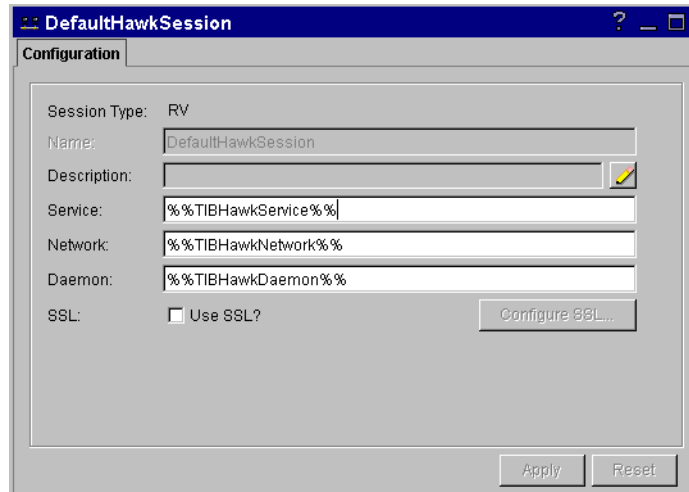
1. In the project tree panel, click the COM Adapter Configuration icon defined for your adapter instance.
2. Select the Show All Tabs check box, then click the Monitoring tab.
3. The Default MicroAgent Session contains the name of the Hawk session:DefaultHawkSession. Use default setting for the other fields.

The next diagram shows the definition for a Monitoring tab.



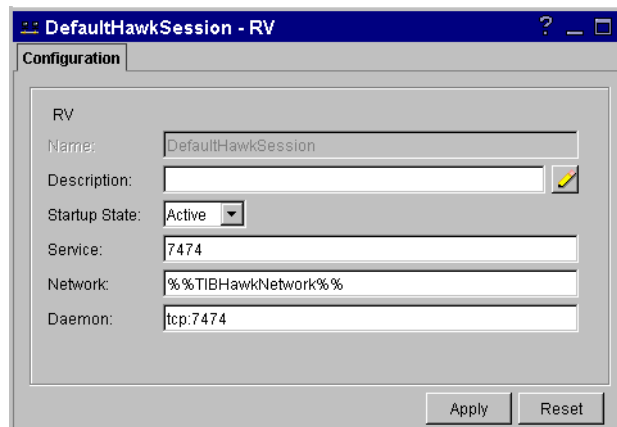
4. Open the Advanced folder for the adapter instance. Double-click the Sessions folder.

5. Double-click the DefaultHawkSession icon.



6. In Service, type **7474** (the default used by TIBCO Hawk) or modify the global variable by clicking the Global Variables tab.
7. In Daemon, type **tcp:7474** (the default used by TIBCO Hawk) or modify the global variable by clicking the Global Variables tab.
8. If you choose to change the defaults, click **Apply** and save the project.

The next diagram shows the HawkSession defined.



Using Global Variables

The variable substitution mechanism can override global variables predefined in the project in a restricted manner. Predefined variables can be viewed and set in TIBCO Designer. Variables are specified as %%VARNAME%% and cannot contain any white space.

Variable substitution allows you to accomplish the following.

- Substitute string variables specified in the project at startup time.
- Locally define the value for a variable for a specific project. The local value takes precedence over any global value.
- Specify the value for a variable in a properties file. This overrides the project repository and values set in code, but not variables set on the command line.
- Enforce the pre-defined variables listed in Predefined Global Variables on page 119.

Variables can be used anywhere in the configuration and will be replaced by the locally-defined adapter instance.

Variable Specification

The adapter can specify variables:

- In the project during configuration using TIBCO Designer
- In a properties file

Properties file values overwrite values set in the project.

Specifying Variables Using TIBCO Designer

Global variables provide an easy way to set defaults for use throughout your project.

For example, you could assign the value 7474 to the predefined global variable `RvDaemon`. You can then use the variable in different sessions in your adapter. If you wish to change the TIBCO Rendezvous daemon for your adapter, you can globally set it to a different value or override it from the command line.

To use global variables in your project, follow these steps:

1. In the project panel, select the `Global Variables` tab.

The project panel is updated to display all currently defined global variables. You now have these choices:

- To assign or change a variable value, select that region and triple-click the variable. The variable expands so you can change either the variable name or the variable value. Press Enter when you're done.
- To add a new global variable group, click the leftmost icon. Specify the name of the group, then press Enter. With the group icon selected, you can click the abc icon to add variables to the group.
- To add a global variable, click the abc icon. A new global variable item is added to the bottom of the list. Supply the variable name and, optionally, the value. Press Enter when you're done.

The global variable is now displayed in the global variables list.

2. When you want to use the global variable in the fields of a resource, enter the variable name surrounded by %% on both sides.

When the project is deployed and the configured components are run, all occurrences of the global variable name are replaced with the global variable value (unless it was overridden in a way that had higher precedence). For example, `RvServiceTest` would be replaced with `7800`.

A number of global variables are predefined. See *Predefined Global Variables* on page 119 for information. You may add definitions of any variables you need to the predefined variables.

Predefined Global Variables

The next table lists and explains the predefined global variables. Some global variables are automatically used within the system when an adapter instance is configured.

Table 13 Predefined Global Variables

Variable	Description
Deployment	Defaults to the TIBCO Designer project name. This value can be any string value. This global variable is used by the system to partially define the subject name defined for a service.
DirLedger	Specifies the path name of the TIBCO Rendezvous certified messaging ledger file. The default is the root installation directory.

Table 13 Predefined Global Variables

Variable	Description
DirTrace	Specifies the path name for log file used by the adapter. The default is the root installation directory.
Domain	The default value for file-based local projects is MyDomain. The value for server-based projects is the domain to which the project was saved.
HawkEnabled	Indicates whether TIBCO Hawk is used to monitor the adapter. True indicates that a Hawk microagent is defined for the adapter. False indicates the microagent is not to be used.
JmsProviderUrl	Specifies where the JMS server is located. Setting this value mostly makes sense in the early stages of a project, when only one JMS server is used.
RemoteRvDaemon	TIBCO Rendezvous routing daemon (rvrd) to be used. See <i>TIBCO Administrator Server Configuration Guide</i> for details about setting up a domain using rvrd.
RvDaemon	TIBCO Rendezvous daemon. Sessions use this daemon to establish communication. The default value is 7500.
RvNetwork	<p>TIBCO Rendezvous network. This variable need only be set on computers with more than one network interface. If specified, the TIBCO Rendezvous daemon uses that network for all outbound messages.</p> <p>In most cases, you can leave the default.</p>
RvService	<p>TIBCO Rendezvous service. The Rendezvous daemon divides the network into logical partitions. Each transport communicates on a single service. A transport can communicate only on the same service with other transports.</p> <p>Unless you are using a non-default TIBCO Rendezvous configuration, you should leave the default (7500).</p>

Table 13 Predefined Global Variables

Variable	Description
RvaHost	Computer on which the TIBCO Rendezvous agent runs. This variable is only relevant if you are using the TIBCO Rendezvous Agent (rva) instead of the TIBCO Rendezvous daemon, and if you have configured a non-default setup. See <i>TIBCO Rendezvous Administration</i> for details about specifying the rva parameters.
RvaPort	TCP port where the TIBCO Rendezvous agent (rva) listens for client connection requests. See <i>TIBCO Rendezvous Administration</i> for details about specifying the rva parameters. Defaults to 7501.
TIBHawkDaemon	TIBCO Rendezvous daemon used in the TIBCO Hawk session. See the <i>TIBCO Hawk Installation and Configuration</i> manual for details about this parameter.
TIBHawkNetwork	TIBCO Rendezvous network used by the TIBCO Hawk session. See the <i>TIBCO Hawk Installation and Configuration</i> manual for details about this parameter.
TIBHawkService	TIBCO Rendezvous service used by the TIBCO Hawk session. See the <i>TIBCO Hawk Installation and Configuration</i> manual for details about this parameter.

Setting Encoding Options

See the *TIBCO Adapter Concepts* book for an introduction to Internationalization topics such as Unicode and how adapters handle it. The encoding for TIBCO Adapter for COM is `UTF16_LittleEndian`.

Complete the following steps prior to running the adapter so it can handle files in different encodings.

1. Configure TIBCO messaging encoding.

The wire format encoding used for communication between adapters and TIBCO-enabled applications is determined by the encoding property set in the project. The adapter instance can be saved in a project:

- At design time or running as a legacy project using a local repository.
- Deployed to a TIBCO Administrator Domain.

If the adapter instance is saved to a project in a Administration server domain, TIBCO messaging encoding is determined by the `repo.encoding` property in the server's `tibcoadmin.tra` file. Each adapter or TIBCO-enabled application that uses the Administration server for storing and retrieving configuration data from a project uses this encoding setting when communicating. This assures that all components (including adapters and other TIBCO-enabled applications) that use the same repository also use the same encoding value to communicate. The `repo.encoding` property value can be `ISO8859-1` (the default) or `UTF-8`. If English or other Latin-1 language data is transmitted between adapters, `ISO8859-1` should be used. Otherwise use `UTF-8`.

If an adapter instance is saved in a local project, the TIBCO messaging encoding is determined by the encoding property of the local project file. To communicate with other adapters using the same encoding, all adapters and applications must set their local project file encoding property to be identical. The encoding value is set on the root project folder, in the `Save Project` dialog box, `TIBCO Messaging Encoding` field. The default encoding is `ISO8859-1`.

The encoding property set in the project file is superseded by the server's encoding property. The encoding property discussed above is the encoding used by the communication between adapters and applications, not the encoding used for the persistent storage of the project files. Project files are always saved using `UTF-8`.

2. The adapter supports automation-compliant BSTR data type for string literals, which is represented internally as a `UNICODE` string. For example, to handle the Euro character (€) correctly, you must set the TIBCO Messaging encoding to `UTF-8`.

Using DCOM

The adapter replaces the RPC Proxy Stub (DCOM)-based access of remote COM servers with TIBCO Rendezvous-based access. Although it is recommended that the service component of the adapter and the COM object run on the same machine, the service component provides the ability to launch components on a remote machine using DCOM.

To use DCOM to launch components on a remote machine using the adapter:

1. Configure and register the DCOM server on a different machine to support remote invocation.
2. Test the server using a COM client from the machine where the service component of the adapter instance will be running.
3. Import the server type library to the repository that the service component instance will be using.



For more information on DCOM configuration, consult MSDN or any introductory text on COM/DCOM.

Using COM+

The adapter supports COM+ and can access the complete functionality of the COM+ runtime environment. It does not require any additional configuration for accessing the COM servers deployed in a COM+ runtime environment.

There are three deployment scenarios which the adapter can be integrated with the COM+ runtime environment, which are:

- The COM server and the service component of the adapter are running on the same machine.
- The COM application is exported as a server application to the machine on which the service component of the adapter is running.
- The COM application is exported as an application proxy to the machine on which the service component of the adapter is running.

If the deployment scenario involves running different components on more than one machine, the scenario should be tested without using the adapter to eliminate any configuration issues.



If the COM+ application containing the server is shut down from Component Services management console, the service component of the adapter might maintain some reference to the objects created in the COM+ runtime environment. Therefore, it is recommended that you restart the service component too.

Configuring Highly Available Instances

You can achieve high availability for the service component of the adapter by running multiple service components as members of a distributed queue group. High availability is typically not a requirement for the interceptor component.

To configure highly available adapter instances, the COM server must be installed on multiple machines, and each of those machines should run service instances that have been configured using an RVCMQ session.

The TIBCO Rendezvous daemon process delivers a message to only one service instance. Therefore, if a service instance goes down, any other instance that is running will receive and process the message. Also, if a service instance is busy processing a message, the TIBCO Rendezvous daemon process will assign the task to another service instance.

To achieve high availability in service components:

1. Build and register the COM server on a machine.
2. Import the metadata into a repository using the `Import Metadata` option on the `Import Schema` tab, in TIBCO Designer. The `Import Schema` tab is available when you select the `Advanced Services` folder in the project panel.

Make sure that you use RVCMQ sessions in the configuration. Run the repository as a remote server.

For more information on configuring the RVCMQ sessions for the service instance of the adapter, see [Configuring the Adapter for Load Balancing Using RVCMQ](#) on page 126.

3. Copy the COM Server executable to other machines. Register the COM server executable in those machines.
4. Run the service instance on each of these machines using the same instance of the remote repository configured in step 2.

Configuring the Adapter for Load Balancing Using RVCMQ

TIBCO Adapter for COM uses the TIBCO Rendezvous-supported RVCMQ sessions to achieve load balancing using distributed queues. Load balancing is achieved at the service component level. To ensure load balancing across machines, the same COM Server must be installed on multiple machines, and each of those machines should run the service instance configured using the RVCMQ session.

The TIBCO Rendezvous daemon process delivers a message to only one service instance. TIBCO Rendezvous balances the load between all the running service instances (RVCMQ Sessions) that share the same CMQ Name. Therefore, when one instance of the service is busy in processing a message, the TIBCO Rendezvous daemon process will assign the task to another service instance. The interceptor component does not directly contribute to the overall load balancing mechanism. However, it should be configured to support load balancing at the service level.

To configure the interceptor and service for load balancing using RVCMQ.

1. Create a project using TIBCO Designer.
2. Use the `Import Metadata` option under the `Import Schema` tab, in TIBCO Designer to import the COM type library information into the project file. The `Import Schema` tab is available when you select the `Advanced Services` folder in the project panel.

Configuring Interceptor Component

To configure the interceptor:

1. Create an adapter instance for the interceptor. You can create a `Request Response Invocation` or `Publication` service that uses the `Certified` quality of service. Specify the subject name on which this service will send the request message.
2. Choose a schema for this service.

Configuring Service Component

To configure the service:

1. Create an adapter instance for the service. You can create a `Request-Response Server` or `Subscription` service that uses the `Distributed` quality of service. Specify the subject name on which this service will listen for messages.

2. Choose a schema for this service.
3. Configure a CMQ session:
 - a. Specify a CMQ Name. Make sure that the CMQ session for all the adapter instances participating in Load Balancing have the same CMQ Name value.
 - b. Specify the Scheduler Activation value. This is a required value that indicates the number of milliseconds within which, if a heartbeat signal is not received by all the queue members, the queue member with the greatest scheduler weight takes its place as the new scheduler. The value of all the queue member sessions should have the same value for this parameter.
 - c. Specify the Scheduler Heartbeat value. This is a required value that indicates the time (in milliseconds) between two heartbeat signals sent by the scheduler. All the queue member sessions should have the same value for this parameter.
 - d. Specify the Scheduler Weight value. This field represents the ability of this session to fulfill the role of a scheduler, relative to other members of the same queue. The higher the weight, the more the precedence for becoming the scheduler.



Changes to the client or the server implementation are not required to use the RVCMQ feature. Configuring the sessions and endpoints will enable the adapter to participate in an RVCMQ environment.

For more information on the RVCMQ/load balancing feature, see the TIBCO Adapter SDK documentation.

Configuring the Adapter for Load Balancing Using JMS

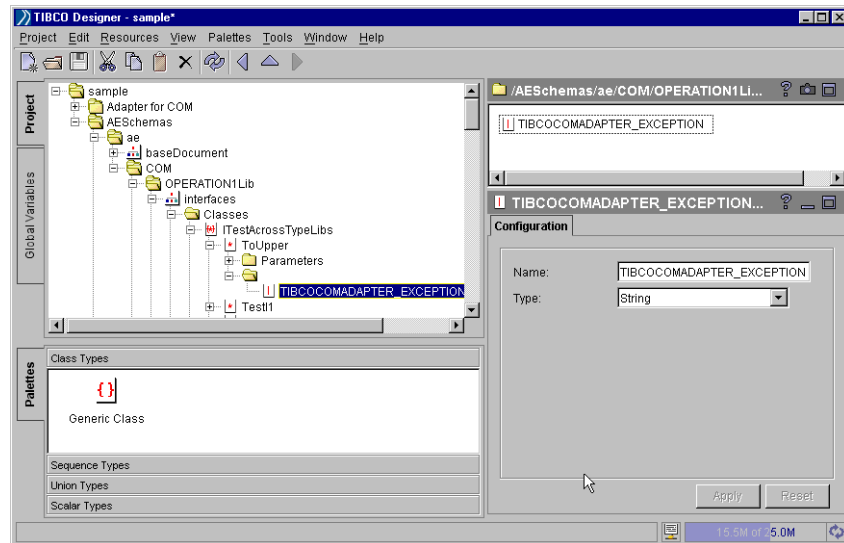
To configure load balancing using JMS:

1. Start the JMS sever.
2. Configure a queue and queue connection factory. Set the `prefetch` property of the queue to a low value (1) so that the load balancing is visible. For non-administered queues (when the queue is created by you and not through the JMS administrator), the default value is 5, so you may need to pump more than five messages to test load balancing.
3. Use the configured queue and queue connection factory to configure the Subscription and Request Response Service in the project.
4. Start many instances of the service. Start a Publication and Request Response Invocation Service to publish data to the same queue. The messages will be distributed across service instances.

For details on queue and queue connection factory, see the *TIBCO Enterprise for JMS* documentation.

Exception Handling

If a two-way operation request fails, a Request Response Service must return a TIBCO ActiveEnterprise exception of some kind. TIBCO ActiveEnterprise exceptions can be defined for each method of a TIBCO ActiveEnterprise class in TIBCO Designer, as shown below.



Every TIBCO ActiveEnterprise Class used by the service component must have the exception `TIBCOCOMADAPTER_EXCEPTION` of type `string` defined for each method in the class. When a COM method invocation returns an unsuccessful `HRESULT` to the service, the service tries to get additional information about the error through the `IErrorInfo` interface, to pass it to the client.

If the COM object did not support the `IErrorInfo` interface or if the service encounters any other kind of error, the service packages the numeric error code, and a textual description of the error (if possible) into a `TIBCOCOMADAPTER_EXCEPTION` and returns this to the Request Response Invocation Service. Other Request Response Services may return exceptions of other kinds.

The `CTestobj1::TestIErrorInfo` function in the `testobj1.cpp` sample file below shows how the Error Information can be set by a COM object, taking advantage of the `IErrorInfo` interface:

```
STDMETHODIMP CTestobj1::TestIErrorInfo(
    /*<in,out>*/ int* iop1)
{
    PRINT_ENTRY("TestIErrorInfo");
    HRESULT hr;
```

```

    if (*iop1 == 1)
    {
        CComPtr<ICreateErrorInfo> spCreateErrorInfo;
        hr = CreateErrorInfo(&spCreateErrorInfo);
        _ASSERT(SUCCEEDED(hr));
        hr = spCreateErrorInfo->SetSource(L"Testobj1");
        _ASSERT(SUCCEEDED(hr));
        hr = spCreateErrorInfo->SetGUID(IID_ITestobj2);
        _ASSERT(SUCCEEDED(hr));
        hr = spCreateErrorInfo->SetDescription(L"An error was
generated.");
        _ASSERT(SUCCEEDED(hr));
        CComPtr<IErrorInfo> spErrorInfo;
        hr = spCreateErrorInfo->QueryInterface(
            IID_IErrorInfo,
            (void*)&spErrorInfo
        );
        _ASSERT(SUCCEEDED(hr));
        hr = SetErrorInfo(0, spErrorInfo);
        _ASSERT(SUCCEEDED(hr));
        hr = 0xE2000004L; // Some error code.
    }
    else if (*iop1 == 2)
    {
        hr = E_INVALIDARG;
    }
    else if (*iop1 == 3)
    {
        hr = 0xFF000004L; // Some error code.
    }
    else
    {
        hr = S_OK;
    }
    PRINT_RETURN("TestIErrorInfo");
    return hr;
}

```


Obtaining Extended Error Information

When the Request Response Invocation Service in the interceptor receives an exception from a Request Response Service (or the interceptor encounters any other kind of exception), it returns one of the following values to the COM client:

```
ERROR_MOPERATIONEXCEPTION_THROWN (0xE2000042)
ERROR_MEXCEPTION_THROWN (0xE2000043)
ERROR_UNKNOWN_EXCEPTION_THROWN (0xE2000044)
```

The interceptor component supports the COM `ISupportErrorInfo` interface and returns `S_OK` to the call `ISupportErrorInfo::InterfaceSupportsErrorInfo`. Thus, when the interceptor returns an unsuccessful `HRESULT` to the COM client, the client should call `GetErrorInfo` on the calling thread to obtain extended error information through the `IErrorInfo` interface. The file `main.cpp` in the `sample.dat` project has sample code to show you how to do this:

Example main.cpp

```
void GetIErrorInfo(
    HRESULT hr
)
{
    CComPtr<IErrorInfo> spErrorInfo;
    ostringstream ss;
    HRESULT hrei = GetErrorInfo(0, &spErrorInfo);
    if (hrei == S_OK)
    {
        CComBSTR Description;
        hrei = spErrorInfo->GetDescription(&Description);
        _ASSERT(SUCCEEDED(hrei));
        ss << "Error encountered ("
            << setbase(16)
            << hr
            << "). "
            << (char*)_bstr_t((BSTR)Description)
            << ends;
    }
    else
    {
        ss << "Error encountered ("
            << setbase(16)
            << hr
            << ")."
            << ends; \
    }
    char* s = ss.str();
    MessageBox(NULL, s, "TIBCOCOMInterceptor", MB_OK);
    delete s;
}
```

Erroneous Method Calls

When the logger component of the adapter receives an error code, it checks the application resource dll (`TIBCOCOMMessage.dll`) as well as the system message-table resource to find out the corresponding string for it. When it fails to format the string, it displays the message, `Could not format error/info message!`

- If a method from the COM server method displays an exception, the service component retrieves it and logs the exception with the error code `ERROR_UNKNOWN_IMPL_EXCEPTION`, with the appropriate coclass, Interface, and method name information.
- If the method from the COM server does not display an exception but returns a failure `HRESULT`, the service component tries to get more information about the failure using the `ISupportErrorInfo` interface. If the COM server does not implement this `ISupportErrorInfo` interface, the service component is unable to get more information about the failure scenario and propagates a `TIBCOCOMException` object with the `HRESULT` code it retrieved from the COM server.
- If the error code that the COM Server returns, due to an invalid operation in the COM server method implementation, is not present in the system message-table resource as well as the adapter message resource, the `Could not format error/info message!` is displayed.

Chapter 8

TIBCO Moniker and Interceptor Activation

This chapter explains how the interceptor component of the adapter uses `tibco moniker`. Although the examples and the explanation in this chapter explicitly use the TIBCO Rendezvous transport, it is applicable to JMS too.

Topics

- *Overview, page 134*
- *Using the TIBCO Moniker in Visual Basic, page 139*
- *Using the TIBCO Moniker in VC++, page 140*
- *Endpoints and COM Coclasses, page 141*
- *Mapping COM and TIBCO ActiveEnterprise Data Types, page 144*

Overview

The interceptor of TIBCO Adapter for COM is similar to a COM proxy. In other words, when a client tries to create an instance of a COM coclass and obtain an interface pointer to it, a proxy is returned to the client. This proxy is an interface pointer with a virtual function table (`vtable`) containing entries that point not to individual method implementations, but rather to generic interception code.

The client uses this proxy as if it were the real interface pointer. However, when the client makes calls to the proxy's virtual functions, the `vtable` reroutes these calls to the generic interception code. This interception code then marshals the method parameters into a TIBCO ActiveEnterprise message and transmits this message over TIBCO Rendezvous.

The mechanism through which the interceptor is activated and a proxy is returned to the client is the TIBCO moniker. The client creates an object using the TIBCO moniker-based syntax.



A moniker is an intelligent class factory used to produce an instance of (or a proxy for) another object. A general description of COM monikers is beyond the scope of this document. For more information on monikers, consult MSDN or an introductory text on COM/DCOM.

Display String Format

The display string of a TIBCO moniker, discussed in more detail in the next section, is divided into two parts by a colon (:). The first part consists of the string `tibco` or `tibcos`. The second part is divided by vertical bars (|) into three required fields and two optional fields, which must appear in the sequence shown here. Thus, the correct format for the display string of a TIBCO moniker is:

```
tibco:<repository_instance_identifier>|<adapter_instance_identifie
r>|<programmatic_identifier><|<override_endpoint>|<properties_file
path>>
```

or

```
tibcos:<repository_instance_identifier>|<adapter_instance_identifi
er>|<programmatic_identifier><|<override_endpoint>|<properties_fil
epath>>
```



The `tibco` form is used to activate the .dll implementation of the interceptor component. The `tibcos` form is used to activate the Microsoft Windows Service implementation of the interceptor component. For more information on the different uses of these two implementations of the interceptor component, see Components on page 2.

Example

An example of a display string for a TIBCO moniker is:
`tibco:tibcr://sample|Adapter for COM/Interceptor
 Instances/TIBCOCOMInterceptor1|Operation2.Testobj1.1.OverrideClient`

Parts of the Display String

The components of the display string are discussed in this section.

Part 1

The first part of a moniker display string always identifies the kind of moniker the COM runtime should activate. In this case, the string `tibco` (or `tibcos`) tells the COM runtime that it should activate a TIBCO moniker (or a TIBCO class factory that will return an interceptor proxy).

Part 2

The second part contains the information the COM runtime is to pass to the TIBCO moniker to tell it how to create the proxy. This information consists of the following items, which must be entered in the command line in sequence.

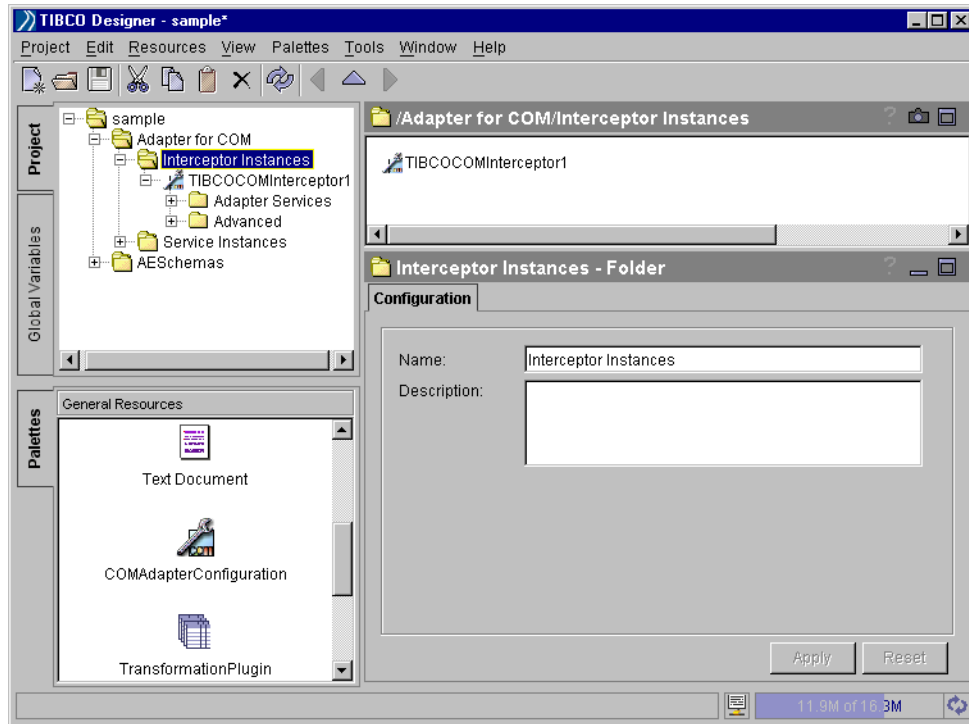
Repository Instance Identifier

The `Repository Instance Identifier` identifies which repository instance contains the configuration information for the interceptor proxy. This identifier takes one of two forms depending on the type of repository the interceptor is using:

- **server-based.** If a server-based repository is used, this identifier takes the form `tibcr://sample`, where `sample` is the name of the repository instance.
- **file-based.** If this repository is used, the identifier takes the form of an absolute file path name, such as
`c:\tibco\adapter\adcom\5.3\examples\sample.dat`.

Adapter Instance Identifier

The Adapter Instance Identifier identifies which adapter configuration instance inside the repository contains the configuration information to be used by the interceptor proxy. When you open a repository instance inside TIBCO Designer, you might see the following tree:



In this case, the Adapter Instance Identifier comprises the nodes:

- Adapter for COM
- Interceptor Instances
- TIBCOCOMInterceptor1

These nodes are concatenated as a file path:

Adapter for COM/Interceptor Instances/TIBCOCOMInterceptor1



You must use a forward slash (/) when you specify the configurl for the TestInterceptor, in the Adapter Properties file.

Programmatic Identifier

The `Programmatic Identifier` is a COM programmatic identifier (progid) used to identify the COM coclass for which you want to create a proxy.

Override endpoint (optional)

`Override endpoint` allows you to bind an interceptor to a specific endpoint (`RequestResponseInvocationServiceEndpoint` or `PublicationServiceEndpoint`) at runtime. The endpoint must be present in the adapter configuration instance in the repository. It is a good practice to explicitly specify the `OverrideEndpoint` in the moniker display name.



When the repository contains multiple endpoints that are associated with the same TIBCO ActiveEnterprise class, and if the `OverrideEndpoint` is not explicitly specified in the moniker display name, the interceptor will choose the first suitable endpoint it finds in the repository. That endpoint may not be the one you intend.

Example 1 In this example, you:

- Define the following two Publication Services in the repository for the adapter configuration instance `TIBCOCOMInterceptor1`:

```
Operation2.Testobj2.1.Publisher
Operation2.Testobj2.1.OverridePublisher
```

- Each of these publication services is associated with the TIBCO ActiveEnterprise class `COM/OPERATION2Lib/coClasses/Operation2.Testobj2.1`.
- This TIBCO ActiveEnterprise class is associated with the TIBCO Adapter for COM coclass identified by the programmatic identifier `Operation2.Testobj2`.

In this example, you can use *either* of the following two moniker display names at runtime:

```
tibco:c:\tibco\adapter\adcom\5.3\examples\sample.dat|Adapter for
COM/Interceptor
Instances/TIBCOCOMInterceptor1|Operation2.Testobj2|Operation2.Test
obj2.1.Publisher
```

or

```
tibco:c:\tibco\adapter\adcom\5.3\examples\sample.dat|Adapter for
COM/Interceptor
Instances/TIBCOCOMInterceptor1|Operation2.Testobj2|Operation2.Test
obj2.1.OverridePublisher
```

If you use the first moniker display name, the interceptor will be bound to the `Operation2.Testobj2.1.Publisher` endpoint. If you use the second moniker display name, the interceptor will be bound to the `Operation2.Testobj2.1.OverridePublisher` endpoint.

Example 2 If the following changes are made to the previous example,

- Do not specify the `OverrideEndpoint` in the moniker:

```
tibco:c:\tibco\adapter\adcom\5.3\examples\sample.dat|Adapter for
COM/Interceptor Instances/TIBCOCOMInterceptor1|Operation2.Testobj2
```

- List the `Operation1.Testobj2.1.OverridePublisher` endpoint before the `Operation1.Testobj2.1.Publisher` in the repository.

As no explicit override endpoint is set in the moniker display name, the adapter uses the `Operation1.Testobj2.1.OverrideEndpoint`, since that is the first endpoint it finds associated with a TIBCO `ActiveEnterprise` class as well as the COM coclass `Operation2.Testobj2`.

Properties filepath (optional)

When the `properties_filepath` field is used, the `override_endpoint` field must either be specified as a value or represented by two bars, as shown:

```
tibco:tibcr://sample|Adapter for COM/Interceptor
Instances/TIBCOCOMInterceptor1|Operation2.Testobj1||c:/propertiesf
ile
```

If you modify the properties file while the interceptor is running, you must release all references to the interceptor for the new values to take effect.

For more information about the properties file, see [Adapter Properties File](#) on page 100.

Using the TIBCO Moniker in Visual Basic

In Visual Basic, the actual operation of the TIBCO moniker is hidden. The client calls the built-in Visual Basic function `GetObject`, and passes the display string to it. `GetObject` creates an instance of the TIBCO moniker (class factory) and then invokes methods on it to obtain the proxy for the class specified in the programmatic identifier. The call to `GetObject` returns a proxy for the default interface of the coclass. For example:

```
Dim Testobj1Var as Testobj1
```

```
Set Testobj1Var = GetObject("tibco:tibcr://sample|Adapter for  
COM/Interceptor  
Instances/TIBCOCOMinterceptor1|Operation2.Testobj1")
```

When the server-based repository needs a username and password to connect, the `GetObject` call should be modified to:

```
Set Testobj1Var =  
GetObject("tibco:tibcr://sample:userName=admin:password=admin|Adap  
ter for COM/Interceptor  
Instances/TIBCOCOMinterceptor1|Operation2.Testobj1")
```

Using the TIBCO Moniker in VC++

In C++, the client calls the Win32 `MkParseDisplayName` function, which returns a moniker. The client then calls the `BindToObject` method on the moniker to obtain the interface pointer (proxy). The `sample.dat` project distributed with the adapter contains the following code:

```
void CreateInstanceThroughMoniker(
    const _bstr_t& RepoURL,    // Repository Instance Identifier
    const _bstr_t& ConfigURL,  // Adapter Instance Identifier
    const _bstr_t& ProgID,     // Programmatic Identifier
    REFIID refiid,            // IID of desired interface (for
                                example, ITestobj1)
    void** ppItf              // Pointer to interface pointer
)
{
    _ASSERT(ppItf);
    *ppItf = NULL;
    CComPtr<IBindCtx> spBindCtx;
    HRXBLOCK(CreateBindCtx(0, &spBindCtx);)
    _bstr_t DisplayName = "tibco";
    DisplayName += ":";
    DisplayName += RepoURL;
    DisplayName += "|";
    DisplayName += ConfigURL;
    DisplayName += "|";
    DisplayName += ProgID;
    ULONG cchEaten;
    CComPtr<IMoniker> spMoniker;
    HRXBLOCK(MkParseDisplayName(spBindCtx, DisplayName, &cchEaten,
                                &spMoniker);)
    HRXBLOCK(spMoniker->BindToObject(spBindCtx, NULL, refiid,
    ppItf);)
}
```

Endpoints and COM Coclasses

TIBCO Adapter for COM relies on a mapping of COM coclasses to TIBCO ActiveEnterprise endpoints and of method definitions in TIBCO ActiveEnterprise classes to their corresponding method definitions in the COM coclasses. For information on TIBCO ActiveEnterprise endpoints, see *TIBCO Adapter SDK Programmer's Guide*.

Method Signatures Must Match

For every TIBCO ActiveEnterprise class method, the COM class default interface must have a corresponding method with the same signature. For example, if TIBCO ActiveEnterprise class has Method1 and Method2, the COM class default interface needs Method1 and Method2. In other words, the pair of methods must:

- Have the same name, for example Method1
- Are of equivalent return type, for example boolean
- Have the same number of parameters
 - Each parameter must have the same name
 - Each parameter must have the same direction (in, out, or in/out)
 - The matching parameters must be of equivalent data types

Request Response Service

A Request Response Service is associated with a subject. It serves as a sink for messages flowing *off* TIBCO Rendezvous.

A Request Response Service is also associated with a TIBCO ActiveEnterprise class that has methods defined for it. For the service component of the adapter, this TIBCO ActiveEnterprise class is associated with the default interface of a particular COM coclass.

When a message arrives over TIBCO Rendezvous for the Request Response Service running inside the service component, the service determines from the message which method of the associated TIBCO ActiveEnterprise class is being called. It then looks up the corresponding method on the default interface of the COM coclass and invokes it.

Request Response Invocation Service

A Request Response Invocation Service is associated with a subject and serves as a source for messages flowing *onto* TIBCO Rendezvous. Just like a Request Response Service, a Request Response Invocation Service is also associated with a TIBCO ActiveEnterprise class that has methods defined for it and this TIBCO ActiveEnterprise class is itself associated with the default interface of a COM coclass. You can use TIBCO Designer to associate the default interface of a coclass with the TIBCO ActiveEnterprise class associated with a Request Response Invocation Service or a Request Response Service. For example, if a Visual Basic client uses the TIBCO moniker to obtain an interceptor proxy and invoke a virtual function through that proxy:

1. The invocation is intercepted by the interceptor.
2. The interceptor determines which COM method was invoked and which method on the TIBCO ActiveEnterprise class the COM method corresponds to.
3. The interceptor then asks the Request Response Invocation Service associated with the TIBCO ActiveEnterprise class to invoke this method on the TIBCO ActiveEnterprise class, causing an operation request to be sent over TIBCO Rendezvous.



You may only associate a TIBCO ActiveEnterprise class with the *default* interface of a COM coclass. This is because TIBCO ActiveEnterprise classes have no notion of multiple interfaces on a single class.

Support for Properties

TIBCO Adapter for COM supports the use of `propput` and `propget` properties. When a COM client accesses a `propput` or `propget` property, the interceptor publishes a corresponding TIBCO Rendezvous Remote Procedural Call (RVRPC) request. Similarly when the service component receives a RVRPC request that corresponds to a COM property, it performs the necessary `propput`/`propget` operations.

Non-COM clients (such as TIBCO Integration Manager or other adapters) can use the `get_` and `put_` operations on the TIBCO ActiveEnterprise class to access the properties of a COM object.

Mapping Properties in the Repository

Properties are mapped to operations in repository by these rules:

- The Name of the property is prepended with a `get_` for `propget` properties.

- The Name of the property is prepended with a `put_` for `propput` properties.
 - For `propput` properties in an `IDispatch`-derived interface, the parameter name is set to be the same as the name of the property.

Publication and Subscription Services

In publication and subscription scenarios where you need to publish or subscribe to a TIBCO ActiveEnterprise message using the adapter, you can configure the adapter services to be of type Publication Service or Subscription Service.

The adapter requires that a COM object used for this purpose implement a single interface, with a single method, that has only one parameter. This parameter corresponds to the TIBCO ActiveEnterprise Class that is being published or subscribed to.

The adapter requires this parameter to be of one of the following types:

- User defined type/Struct (UDT)
- Variant that contains a UDT
- Variant that contains a Safearray of UDTs
- Safearray of UDTs

The parameter also has to be a unidirectional `<in>` parameter.



IN/OUT and OUT parameters are not supported.

Mapping COM and TIBCO ActiveEnterprise Data Types

Parameters of TIBCO ActiveEnterprise methods and COM methods must be of equivalent types. Likewise, if a single parameter of a COM method is associated with a Publication or Subscription Service, the COM type of this parameter must correspond to a TIBCO ActiveEnterprise type.

The table below shows the mapping between COM (or VB types) and TIBCO ActiveEnterprise types:

Table 14 COM and TIBCO ActiveEnterprise Data Types

COM Type	TIBCO ActiveEnterprise Type
VT_I1	i1
VT_UI1	ui1
VT_I2	i2
VT_UI2	ui2
VT_I4	i4
VT_UI4	ui4
VT_INT	i4
VT_UINT	ui4
VT_BOOL	boolean
VT_R4	r4
VT_R8	r8
VT_CY	fixed.19.4
VT_DATE	dateTime
VT_BSTR	string or char. <i>n</i> ^a
User Defined Type (UDTs)	class
SAFEARRAY of x	sequence<x>

a. *n* represents the maximum number of characters allowed in a string.

For more information on the mapping between TIBCO ActiveEnterprise and COM data types, see Appendix A, Data Types.

This chapter uses the demonstration programs shipped with TIBCO Adapter for COM to illustrate how to set up and use the adapter. Demonstrations for C++, Visual Basic, and Active Server Page are also provided.

Topics

- *Overview, page 148*
- *C++ Demonstration Programs, page 150*
- *Visual Basic Demonstration Programs, page 157*
- *Active Server Page (ASP) Demonstration: ASPDemo, page 161*
- *Bridging between TIBCO Messaging Transport and BizTalk, page 163*
- *MSMQ Demonstration Example, page 176*
- *.NET Remoting Examples, page 179*

Overview

TIBCO Adapter for COM ships with local repository instances and demonstration projects that have all the configuration information required to run the demonstration programs. The sample repository instances and demonstration projects are located in the directory `<adcom_install_dir>\5.3\examples`.

Repository Instances

The following local repository instances are shipped with the adapter:

- `sample.dat` — A local repository configured to send operations and documents over TIBCO Rendezvous.
- `sampleCm.dat` — A local repository configured to send operations and documents over TIBCO Rendezvous using certified message delivery.

Demonstration Projects

All the demonstration projects are hard-coded to use the local repositories in the directory `c:\tibco\adapter\adcom\5.3\examples\`. If you did not install the adapter to the default installation directory, you will have to change each project's file path to the repository.



All the demonstration projects depend on the `operation` sample, so build it first.

During compilation, all sample code supplied with the adapter installation compile using the default Visual Studio 6.0 installation. If any of the sample code does not compile, check the Visual Studio environment path variables. Make sure that the default environment path takes precedence over other variable paths.

The demonstration projects are:

- **operation** — An in-process COM Server written in C++. This project includes three COM coclasses:
 - `Testobj1` demonstrates how the interceptor can convert COM method invocations into TIBCO ActiveEnterprise operation requests and how the service can convert TIBCO ActiveEnterprise operation requests into COM method invocations.
 - `Testobj2` demonstrates how the interceptor can convert COM method invocations into published TIBCO ActiveEnterprise documents and how

the service can convert published TIBCO ActiveEnterprise documents into COM method invocations.

- `Testobj3` also demonstrates how the interceptor can convert COM method invocations into published TIBCO ActiveEnterprise documents and how the service can convert published TIBCO ActiveEnterprise business documents into COM method invocations.

With the respective default interfaces:

- `ITestobj1`
- `ITestobj2`
- `Testobj3`

- **testinterceptor** — A COM client application written in C++. This sample shows how a C++ client uses `Testobj1`, `Testobj2` and `Testobj3` to send operations/documents through the interceptor using TIBCO Rendezvous. This project uses the local repository `sample.dat`.
- **testinterceptorCm** — A COM client application written in C++. This sample shows how a C++ client can use `Testobj1`, `Testobj2` and `Testobj3` to send operations/documents through the interceptor using TIBCO Rendezvous certified messaging. This project uses the local repository `sampleCm.dat`.
- **VBOperationDemo** — A simple COM client application written in Visual Basic. This sample shows how a Visual Basic client can use `Testobj1` to send an operation request through the interceptor using TIBCO Rendezvous. This project uses the local repository `sample.dat`.
- **VBPubSubDemo** — A simple COM client application written in Visual Basic. This sample shows how a Visual Basic client can use a third COM object, `Testobj3`, to publish UDT or SAFEARRAY variables through VARIANT parameters. This project uses the local repository `sample.dat`.
- **ASPDemo** — A simple Active Server Page demo. This sample shows how an Active Server Page can use `Testobj1` to send an operation request through the interceptor using TIBCO Rendezvous. This project uses the local repository `sample.dat`.

C++ Demonstration Programs



All the demonstration projects depend on the operation sample, so build it first.

operation

The operation sample consists of two Microsoft Visual C++ ATL App Wizard projects, `operation1` and `operation2`. The `operation1` project simply defines various UDTs and interfaces. The `operation2` project defines coclasses that utilize the UDT and interface definitions from `operation1`. The `operation2` project contains the following three coclasses:

- `Testobj1`
- `Testobj2`
- `Testobj3`

With the corresponding default interfaces from the `operation1` project:

- `ITestobj1`
- `ITestobj2`
- `ITestobj3`

1. Open the `operation2` project in the `examples\operation\operation2` directory.
2. Build the `operation2` project, causing the `operation2.dll` to be generated and registered in the registry.

You may use the operation sample to provide you with guidance on how to construct parameters of exceptionally complex data types.

C++ Client Projects

The service component is not the only Request Response Service or Subscription Service that can process operation request/documents published by `testinterceptor`. Any Request Response Service or Subscription Service can perform this processing for `testinterceptor`. Using the service component, however, provides an easy way to verify the operation of the adapter.



These exercises assume that you will use `sample` as the name of the dat file, and also assume that you will export `sample.dat` to the directory `C:\tibco\adapter\adcom\5.3\examples`. If you export it to a different directory, you must update the path in `testinterceptor.dsw` and rebuild your COM client before you import the metadata into the project file.

testinterceptor

This is an MSVC++ Win32 Console Application.

1. Open the `testinterceptor.dsw` project in the `examples\TestInterceptor` directory. Set the `testinterceptor` project to active.
2. Build the `testinterceptor` project, to generate the `testinterceptor.exe` executable.
3. Start the service component of the adapter by typing the following line at the Microsoft Windows command prompt:

```
adcomservice -system:repoURL
"c:\tibco\adapter\adcom\5.3\examples\sample.dat"
-system:configURL "Adapter for COM/Service
Instances/TIBCOCOMService1"
```

4. Run `testinterceptor.exe`. It will perform the following actions:
 - a. Call `CreateInstanceFromMoniker` to obtain the `ITestobj1`, `ITestobj2` and `ITestobj3` interfaces. These interfaces are actually interceptor proxies.
 - b. Call methods on the `ITestobj1`, `ITestobj2` and `ITestobj3` interfaces. These methods will result in TIBCO ActiveEnterprise operation requests or published documents being sent over TIBCO Rendezvous. These operation requests/documents will be received and processed by the service component.



This example uses TIBCO Rendezvous. If you wish to use JMS, change the transport type to JMS.

testinterceptorCm

This is an MSVC++ Win32 Console Application.

1. Open the `testinterceptor.dsw` project in the `examples\TestInterceptor` directory. Set the `testinterceptorCm` project to active.
2. Build the `testinterceptorCm` project, causing the executable `testinterceptorCm.exe` to be generated.
3. Start the service component of the adapter by typing the following line at the Microsoft Windows command prompt:

```
adcomservice -system:repoURL
"c:\tibco\adapter\adcom\5.3\examples\sampleCm.dat"
-system:configURL "Adapter for COM/Service
Instances/TIBCOCOMService1"
```

4. Run `testinterceptorCm.exe`. It will perform the following actions:
 - a. Call `CreateInstanceFromMoniker` to obtain the `ITestobj1`, `ITestobj2` and `ITestobj3` interfaces. These interfaces are actually interceptor proxies.
 - b. Call methods on the `ITestobj1`, `ITestobj2` and `ITestobj3` interfaces. These methods will result in TIBCO ActiveEnterprise operation requests or published documents being sent over TIBCO Rendezvous using certified messaging. These operation requests/documents will be received and processed by the service component.

RecordSet Example

This section explains how to use the `testinterceptor` program (that ships with the example code), to test the ADODB RecordSet support provided by the adapter. This example uses the Microsoft SQL Server 2000 for the database support.

A sample server source code written in C++, which interacts with the database, is provided in the `examples\Operation\Operation3` directory of the adapter installation.

The example in `\Operation\Operation3` contains a `OPERATION3` module that defines a class `Testobj4`.

The `Testobj4` class contains only one method named `GetRecordSet`, which implements the functionality provided by the class. The `GetRecordSet` method of `Testobj4` class takes two arguments and returns a ADODB RecordSet pointer. The first argument is the connection string and the second argument is the SQL query string.

The format of the connection string is similar to the following:

```
"user id= <User_id>;Provider=SQLOLEDB;Data Source=
<Server_name>;Initial Catalog= <Database_name>;"
```

The format of the query string is similar to the following:

```
"Select <column_names> from <table_name> where <conditions>"
```

The TestInterceptor application is enhanced to contain a method called TestRecordSet that invokes TestObj4::GetRecordSet() when TEST_RECORDSET is defined in the project. Also, the project file Sample.dat, available in the examples directory contains additional metadata for the Operation3::TestObj4 class, and associated endpoint configurations.

To test the ADO DB RecordSet support using the testinterceptor program:

1. The TestRecordSet.sql available in the examples\Operation\Operation3 directory is an SQL script that creates a table and inserts some test records in that table.

Run the Query Analyzer, open the SQL script, and execute that query.

Also, you can run the query from the command prompt using the isqlw utility. For more information on isqlw, see the documentation that ships with Microsoft SQL Server 2000.

2. Open and build the Operation\Operation3 project from the examples directory. Make sure that Operation3.dll is registered with the system.
3. Open the TestInterceptor project using Microsoft Visual Studio. Modify the Connectionstring parameter for the database connection in the TestRecordSet method. Provide the User ID, Provider, Data Source, and Initial Catalog values according to your database server settings. By default, the recordset method call is not enabled in the project. Comment out the "#define TEST_RECORDSET" line to call the TestRecordSet method.

Connection Object Example

This section explains how the TIBCO ActiveEnterprise client (TIBCO BusinessWorks example) uses the ADO connection object implemented in the COM server.

This example uses Microsoft SQL Server 2000 for the database support.

COM Server Example in Microsoft VC++

A sample server source code written in C++, which interacts with the database, is available at the `examples\TEAKExamples\ConnectionExample\VCExample\ConnectionServer` directory of the adapter installation.

The COM server exposes an interface `ITestConnection` which has the following method:

1. `GetConnection([in] BSTR szConnectionString, [in,out] _Connection** pConn);`

This method is called by the Client to get the Connection pointer from the server, by passing a valid connection string as the `[in]` parameter. The second parameter can be of type `[in, out]`, `[out, retval]`.

2. `GetData([in] BSTR szQueryString, [in, out] _Connection** pConn, [out, retval] _Recordset** pRecSet);`

This method is called by the Client to get a recordset by passing the connection pointer available from the `GetConnection` method and providing a proper query string as the `[in]` parameter. The second parameter can be of type `[in]` or `[in,out]`.

3. `ReleaseConnection([in]_Connection* pConn);`

This method is called by the Client to release the connection object obtained from the `GetConnection` method.

COM Server Example for Microsoft Visual Basic

A sample server source code written in Microsoft Visual Basic, which interacts with the database, is available at the `examples\TEAKExamples\ConnectionExample\VBExample\ConnectionServer` directory of the adapter installation.

Its interface and method implementations are similar to the C++ COM server, but is written in Microsoft Visual Basic.

TIBCO ActiveEnterprise Client Example for C++ Server

Three TIBCO BusinessWorks process scenarios are provided as client examples. The repository file `[RepoDir]` is available at `examples\TEAKExamples\ConnectionExample\VCExample`. The corresponding dat file `[repdata.dat]` is available at `examples\TEAKExamples\ConnectionExample\VCExample`.

The project files can be opened in TIBCO Designer.

The three process scenarios are:

- `Simple Process` describes one connection, and data retrieval using that connection, and finally release of that connection. All the three implemented methods in the Example COM server are used in this TIBCO ActiveEnterprise process example.
- `OneConnectionOnMultipleRecordset` process explains the possible usage of Connection implementation. The client can use the same connection object to get multiple recordsets, that is, data retrieval .
- `MultipleConnection` process explains the scenario of multiple connection objects that can be created, and their data retrieval.

Running Connection Object Examples

1. In each of the three example process scenarios, the Connection string input value of the Connection Activity, that is, the Activity which represents the `GetConnection` method, is given as `DSN=AdoDemo;UID=sa;PWD=;`.
You must create a valid DSN by clicking **Control Panel>Administrative Tools> Data Sources (ODBC)**, and change the connection string accordingly.
2. The `TestRecordSet.sql` available in the `examples\Operation\Operation3` directory is an SQL script that creates a table and inserts some test records in that table. Run the SQL Query Analyzer, open the SQL script, and execute that query.
Also, you can run the query from the command prompt using the `isqlw` utility. For more information on `isqlw`, see the documentation that ships with Microsoft SQL Server 2000.
3. Open Microsoft Visual Studio and build the COM server available at `TEAExamples\ConnectionExample\VCExample\ConnectionServerproject`. Make sure that `ConnectionServer.dll` is registered with the system.
4. Open the Multi-file project in TIBCO Designer and run the process scenarios.

TIBCO ActiveEnterprise Client Example for Visual Basic COM Server

The description and instructions for running this example are similar to the TIBCO ActiveEnterprise Client Example for C++ Server on page 154. The example is available at `examples\TEAExamples\ConnectionExample\VBExample`.

C++ Troubleshooting

Problem: You may get the following error from either the service or interceptor component:

Error encountered (e2000042). MException thrown: Connection to TIB CR failed. Instance name attempted to is
c:\tibco\adapter\adcom\5.3\examples\sample.

Solution: You probably installed TIBCO Adapter for COM in a directory other than the default (c:\tibco\adapter\adcom). To allow the interceptor to find the repository, you need to edit the file main.cpp in the testinterceptor project and change all occurrences of the string
c:\tibco\adapter\adcom\5.3\examples\sample.dat to point to the correct location of the repository file. To allow the service to find the repository, you must provide the correct value for the system:repourl command line switch.

Visual Basic Demonstration Programs



If you have a Visual Basic project group containing both the COM client and the server, the interceptor component might not work properly if you try to debug the projects from inside the Visual Basic project group. Separating the client and the server to different Visual Basic projects, or running the executables with the adapter should resolve this scenario.

These exercises assume that you will use `sample` as the name of the dat file, and also assumes that you will save `sample.dat` to the directory `C:\tibco\adapter\adcom\5.3\examples`. If you save it to a different directory, you must update the path in the `testinterceptor.dsw` and rebuild your COM client before you import the metadata into the project file.

VBOperationDemo

This is a Visual Basic Application.

1. Open the VBOperationDemo project in the `examples\Demos\VBOperationDemo` directory.
2. Build the VBOperationDemo project, causing the executable `VBInterceptorDemo.exe` to be generated.
3. Start the service component of the adapter.
4. Run VBOperationDemo.exe. It will perform the following actions:
 - a. Call `GetObject` to obtain an `ITestobj1` interface. Recall that this interface is actually an interceptor proxy.
 - b. Call the method `ToUpper` on the `ITestobj1` interface. This method will result in a TIBCO ActiveEnterprise operation request being sent over TIBCO Rendezvous. This operation request will be received and processed by the service component.

Code

All the code for the VBOperationDemo project is listed below:

```
Dim objTestobj1 As Testobj1
Private Sub Form_Unload(Cancel As Integer)
Set objTestobj1 = Nothing
End Sub
```

```

Private Sub Form_Load()
Set objTestobj1 =
GetObject("tibco:c:\tibco\adapter\adcom\5.3\examples\sample.dat|Ad
apter for COM/Interceptor
Instances/TIBCOCOMInterceptor1|Operation2.Testobj1")
End Sub

Private Sub Invoke_Click()
If Len(Text1.Text) Then
    Let Text2.Text = objTestobj1.ToUpper(Text1.Text)
End If
End Sub

```

VBPubSubDemo

This is a Visual Basic Application.

1. Open the VBPubSubDemo project in the examples\Demos\VBPubSubDemo directory.
2. Build the VBPubSubDemo project, causing the executable VBPubSubDemo.exe to be generated.
3. Start the service component of the adapter.
4. Run VBPubSubDemo.exe. It will perform the following actions:
 - a. Call GetObject to obtain an ITestobj3 interface. Recall that this interface is actually an interceptor proxy.
 - b. Call the method TestPublishComplexTypeInVARIANT three times when you click the **Send** button, passing a UDT and a couple of SAFEARRAY's through the VARIANT parameter ip1. This method will result in the UDT and SAFEARRAY's being published by the Publication Service in the Interceptor and received by the Subscription Service in the service. The UDT will be published as a TIBCO ActiveEnterprise MInstance, while the SAFEARRAY's will be published as TIBCO ActiveEnterprise MSequences.



It is not possible in Visual Basic to pass a UDT or a SAFEARRAY by value, that is, as an <in> parameter. Visual Basic requires that UDTs and SAFEARRAY's be passed by reference, that is, as <in, out> parameters. Passing <in, out> parameters, however, runs contrary to the entire notion of publish/subscribe, which is one-way in nature.

This project shows how to overcome this problem. You pass the UDT or SAFEARRAY inside a VARIANT, which Visual Basic does allow to be passed by value (as an <in>) parameter. This project also shows how you can pass a variety of documents (MInstances or MSequences) through a single method: simply pass them inside a VARIANT.

Code

All the code the for the VBPubSubDemo project is listed below:

```
Dim objTestobj3 As Testobj3
Private Sub Send_Click()

    ' Publish a UDT in a VARIANT
    Dim varUDTSimple1 As OLEAutomationCompatibleUDTSimple
    Let varUDTSimple1.m_vui1 = 1
    Let varUDTSimple1.m_vi2 = 2
    Let varUDTSimple1.m_vi4 = 4
    Let varUDTSimple1.m_vint = 4
    Let varUDTSimple1.m_vboolean = True
    Let varUDTSimple1.m_vr4 = 4.4
    Let varUDTSimple1.m_vr8 = 8.8
    Let varUDTSimple1.m_vfixed = 1.1
    Let varUDTSimple1.m_vdateTime = Now
    Let varUDTSimple1.m_vstring = "abc"
    Call objTestobj3.TestPublishComplexTypeInVARIANT(varUDTSimple1)

    'Publish a SAFEARRAY of Floats in a VARIANT
    Dim varLongArray(1) As Single
    Let varLongArray(0) = 1.1
    Let varLongArray(1) = 2.2
    Call objTestobj3.TestPublishComplexTypeInVARIANT(varLongArray)

    'Publish a SAFEARRAY of UDT's in a VARIANT
    Dim UDTSimpleArray(1) As OLEAutomationCompatibleUDTSimple
    Let UDTSimpleArray(0).m_vui1 = 1
    Let UDTSimpleArray(0).m_vi2 = 2
    Let UDTSimpleArray(0).m_vi4 = 4
    Let UDTSimpleArray(0).m_vint = 4
    Let UDTSimpleArray(0).m_vboolean = True
    Let UDTSimpleArray(0).m_vr4 = 4.4
    Let UDTSimpleArray(0).m_vr8 = 8.8
    Let UDTSimpleArray(0).m_vfixed = 1.1
    Let UDTSimpleArray(0).m_vdateTime = Now
    Let UDTSimpleArray(0).m_vstring = "abc"
    Let UDTSimpleArray(1).m_vui1 = 1
    Let UDTSimpleArray(1).m_vi2 = 2
    Let UDTSimpleArray(1).m_vi4 = 4
    Let UDTSimpleArray(1).m_vint = 4
    Let UDTSimpleArray(1).m_vboolean = True
    Let UDTSimpleArray(1).m_vr4 = 4.4
    Let UDTSimpleArray(1).m_vr8 = 8.8
    Let UDTSimpleArray(1).m_vfixed = 1.1
    Let UDTSimpleArray(1).m_vdateTime = Now
    Let UDTSimpleArray(1).m_vstring = "abc"
    Call objTestobj3.TestPublishComplexTypeInVARIANT(UDTSimpleArray)

End Sub
Private Sub Form_Load()
```

```
Set objTestobj3 =  
GetObject("tibco:c:\tibco\adapter\adcom\5.3\examples\sample.dat|Ad  
apter for COM/Interceptor  
Instances/TIBCOCOMInterceptor1|Operation2.Testobj3")  
  
End Sub  
Private Sub Form_Unload(cancel As Integer)  
  
Set objTestobj3 = Nothing  
  
End Sub
```

Active Server Page (ASP) Demonstration: ASPDemo

This demonstration consists of a couple of Active Server Pages that accept a string as input, convert it to upper case, and display the resulting string. This is accomplished by a call to the `ToUpper` method on the `ITestobj1` interface.

The ASP program consists of the following files:

- `RequestForm.asp`
- `ResultForm.asp`

In addition to TIBCO Adapter for COM, you must have either the Microsoft Internet Information Server or the Microsoft Personal Web Server installed and running.

Code

The following is the code for `RequestForm.asp`:

```
<%@ Language=VBScript %>
<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<BODY>

<P>&nbsp;</P>

<CENTER>
<H1> Method Invocation using the COM Adapter</H1>
<FORM NAME="RequestForm" ACTION="ResultForm.asp" METHOD=POST>
  Enter the String To Be Upper Cased <INPUT TYPE=TEXT
NAME="ToUpperString"><BR><BR>
  <INPUT TYPE=SUBMIT VALUE="Upper">
  <INPUT TYPE=RESET VALUE="Clear">
</FORM>
</CENTER>

</BODY>
</HTML>
```

The following is the code for `ResultForm.asp`:

```
<%@ Language=VBScript %>
<HTML>
<HEAD>
<TITLE> Result Of Invoking ToUpper Method </TITLE>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<BODY>
<CENTER>
```

```

<H1> Result Of Invoking the ToUpper Method</H1>

<%
Set objTestobj1 =
GetObject("tibco:c:\tibco\adapter\adcom\5.3\examples\sample.dat|Ad
apter for COM/Interceptor
Instances/TIBCOCOMInterceptor1|Operation2.Testobj1")
Dim vBstrIn, vBstrRet
vBstrIn = Request.Form("ToUpperString")
vBstrRet = objTestobj1.ToUpper(vBstrIn)
%>
<B> String To Be Upper Cased : </B> <%=vBstrIn%><BR><BR>
<B> Upper Cased String is : </B> <%=vBstrRet%><BR>
<CENTER>
</BODY>
</HTML>

```


Bridging between TIBCO Messaging Transport and BizTalk

TIBCO Adapter for COM can be used as a bridge between the TIBCO messaging infrastructure and Microsoft's BizTalk Server. TIBCO messages can flow from the TIBCO messaging transport to BizTalk, and BizTalk messages can flow from BizTalk to the TIBCO messaging transport.

This example is developed to work with Microsoft BizTalk Server 2000 and Microsoft Windows 2000. If you want to run this example using Microsoft BizTalk Server 2004, you will have to write a custom component that will help Microsoft BizTalk Server 2004 interface with the adapter.

To illustrate this functionality, this section provides a simple example of a BizTalk schedule that receives messages from the TIBCO messaging transport and republishes them to the TIBCO messaging transport on a different subject.



To keep it simple, the interface/message format for both incoming and outgoing messages are the same, although they can be different.

In this example, you will create a TIBCO Adapter for COM interface that defines the format of messages and then set configuration parameters in BizTalk Orchestration Designer and TIBCO Designer.

The tasks are:

1. Define the COM interface.
2. Pass Messages to BizTalk.
3. Pass BizTalk Messages to TIBCO Messaging Transport.

Define the COM interface

1. Use MSVC++ or VB to create a COM coclass named `Comp` with an interface, `IComp`, that satisfies the following IDL:

```
<
    uuid(44EC00B5-5786-40AB-A26F-6ACAE1EC09B7),
    version(1.0),
    helpstring("QComp 1.0 Type Library")
>
library QCOMPLib
{
    importlib("stdole2.tlb");

    interface IComp;

<
    uuid(90C89293-9CE6-44BA-AC1D-8977FB6C4618),
```

```

        helpstring("Comp Class")
    >
    coclass Comp {
    <default> interface IComp;
};

<
odl,
uuid(B72F57AD-9358-404E-BFC6-19EF1EF0818D),
helpstring("IComp Interface"),
dual,
oleautomation
>
interface IComp : IDispatch {
    <id(0x00000001), helpstring("method M1")>
    HRESULT M1(
        <in> unsigned char vui1,
        <in> short vi2,
        <in> unsigned short vui2,
        <in> long vi4,
        <in> unsigned long vui4,
        <in> int vint,
        <in> unsigned int vuint,
        <in> VARIANT_BOOL vboolean,
        <in> single vr4,
        <in> double vr8,
        <in> CURRENCY vfixed,
        <in> DATE vdateTime,
        <in> BSTR vstring);
    };
};

```

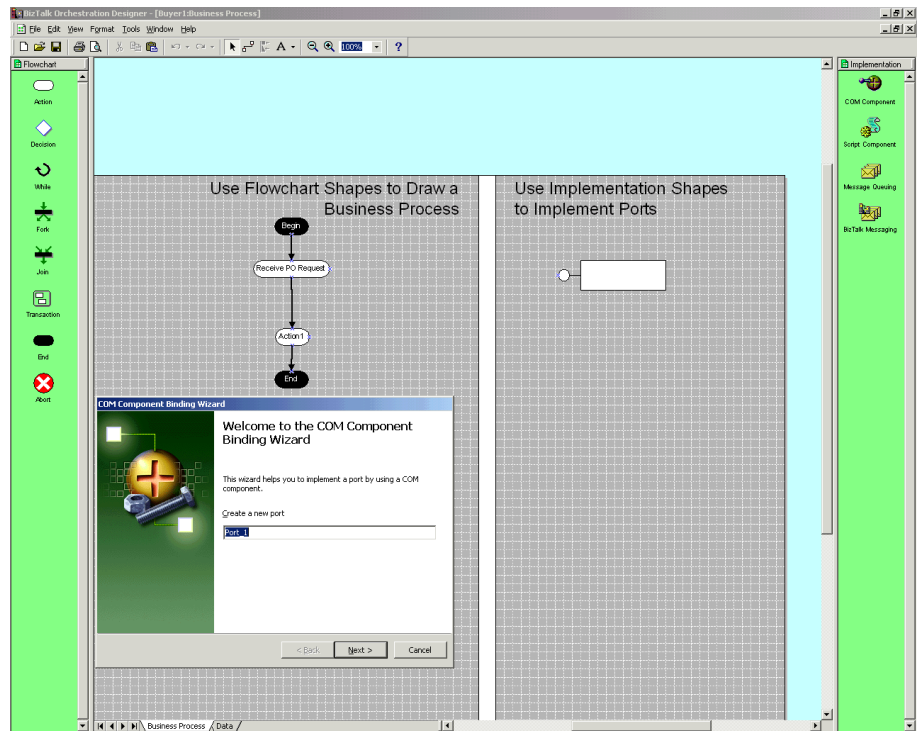
Pass Messages to BizTalk

To allow messages to pass from the TIBCO messaging transport to BizTalk, you must create an incoming COM component port in BizTalk and then configure the service component of the adapter to write messages into the port.

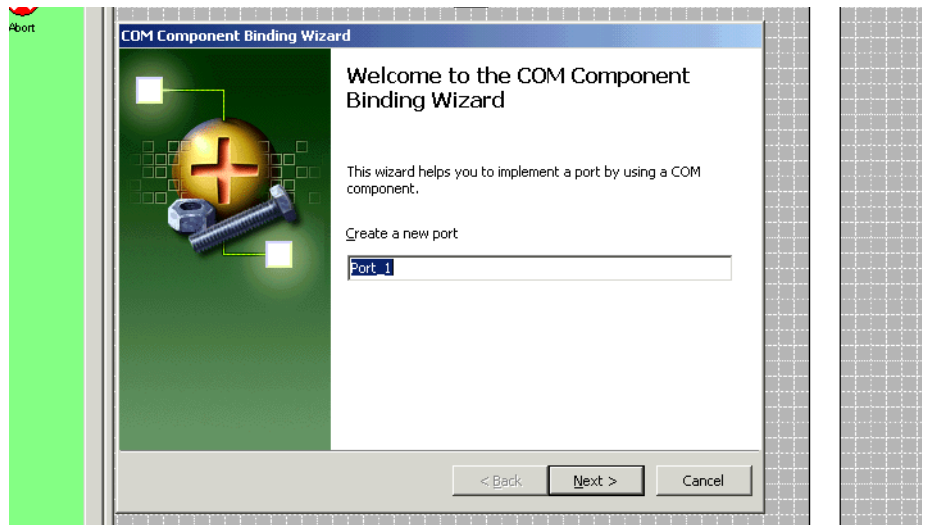
Creating an incoming "COM component" port in BizTalk

1. Start the BizTalk Orchestration Designer.
2. Open the .skv file and select the **Business Process** tab.

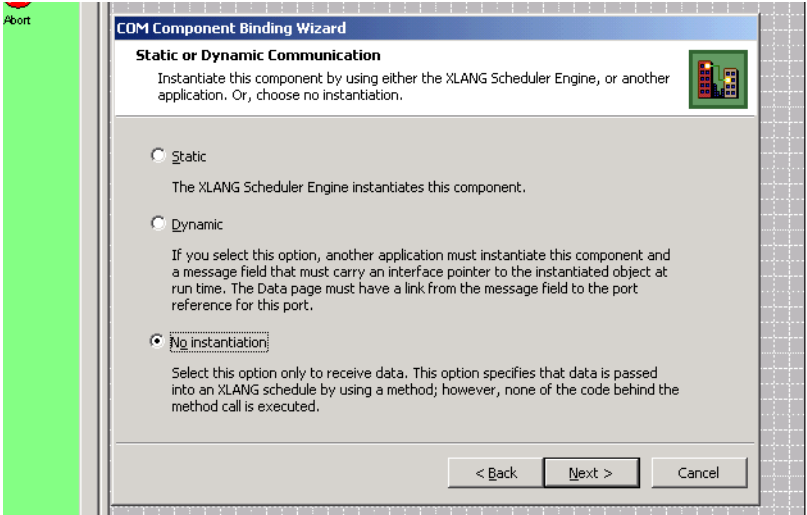
3. From the Implementation palette, drag a COM component port into the right hand panel. This displays the COM Component Binding Wizard.



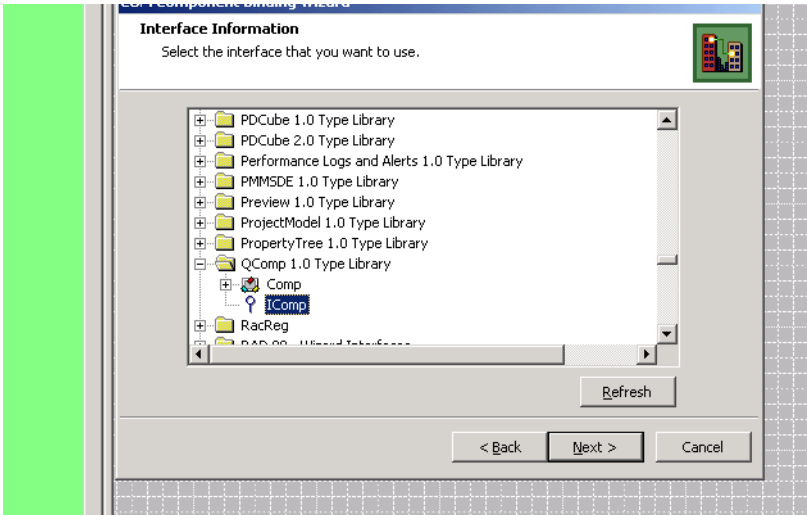
4. Enter a name for the port and click Next.



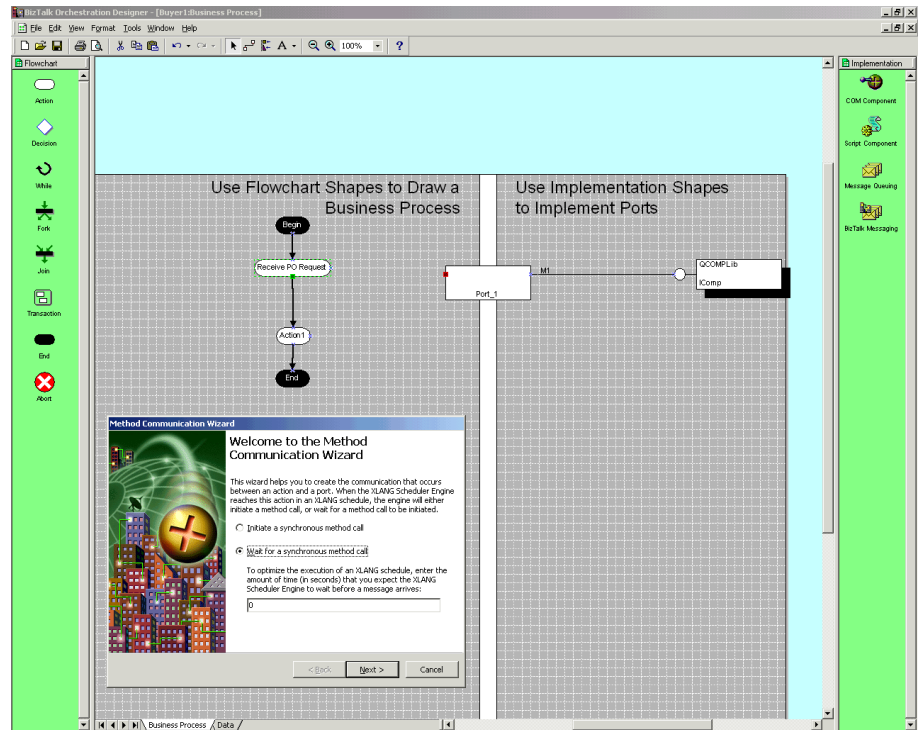
5. On the Static or Dynamic Communication page, select the **No Instantiation** radio button and click **Next**.



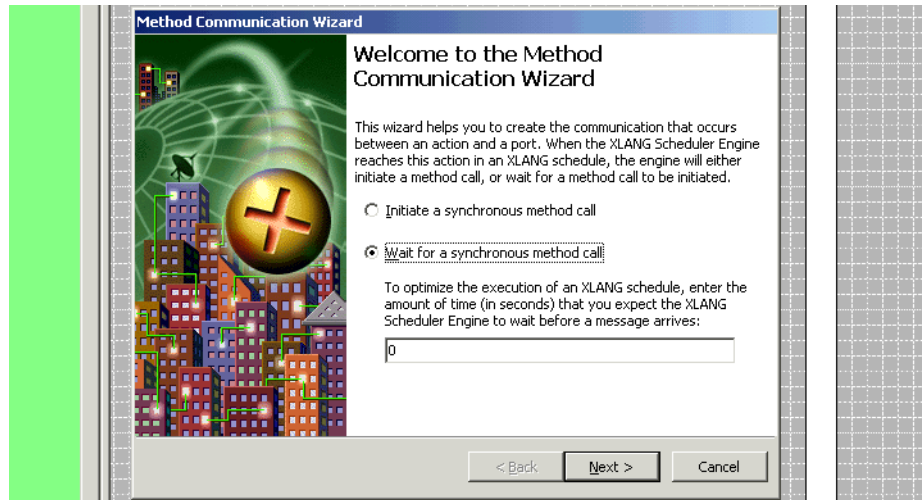
6. On the Interface Information page, select the **IComp** interface to associate it with the port and click **Next**. The COM Component Binding Wizard closes.



7. Drag the right-hand connector from an **Action** shape to the left-hand connector of the port. This displays the Method Communication Wizard.

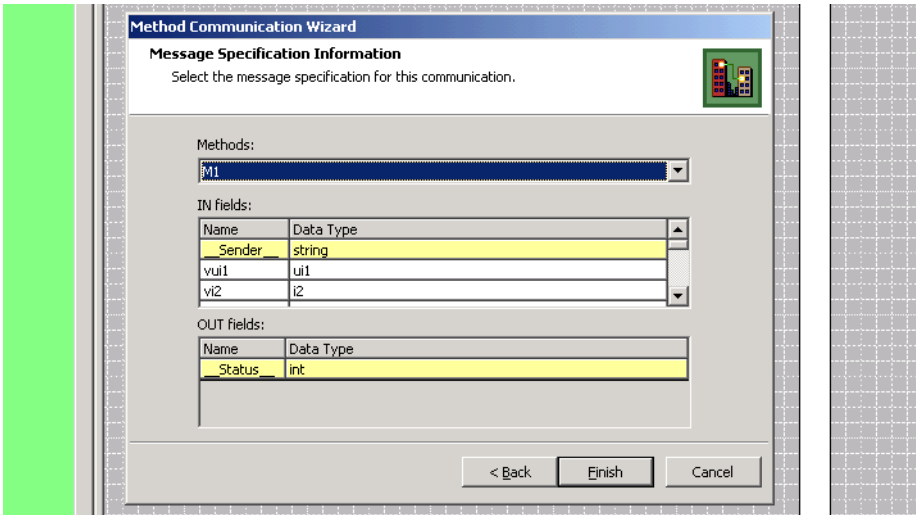


8. Select the **Wait for a synchronous method call** radio button and click **Next**.

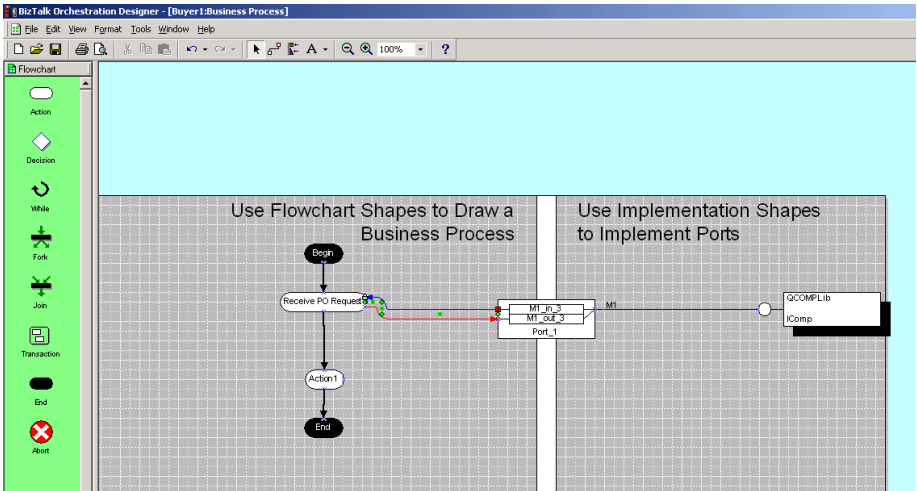


9. Click **Next** to skip the Message Information page.

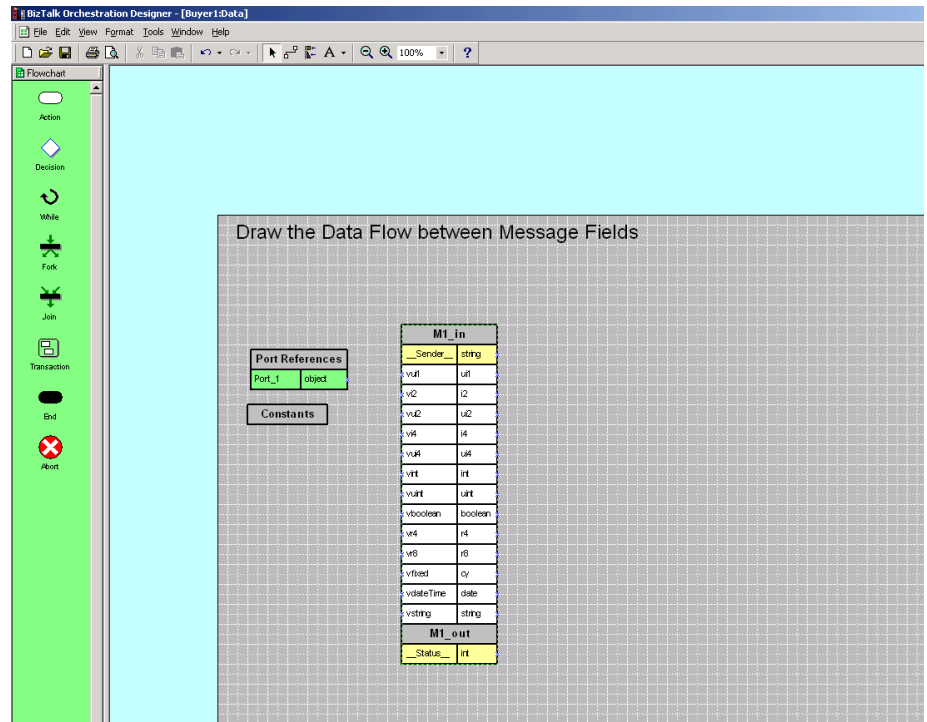
10. On the Message Specification Information page, select the **M1** method on the COM interface to define the BizTalk message format. The type information from the parameters of the method is used to define the message format.



11. Click **Finish**. The **Action** shape is now connected to the incoming port.



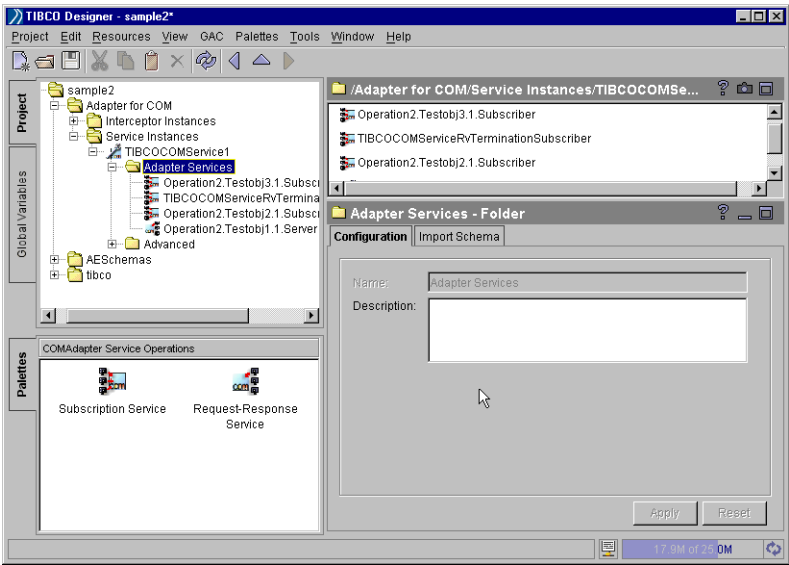
12. If you select the Data tab in the Orchestration Designer, you will see the message format displayed.



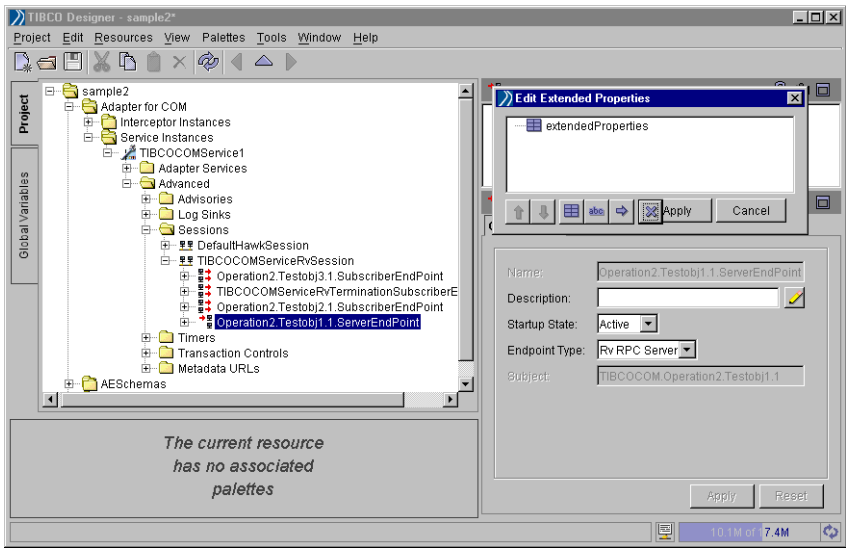
13. Compile the .skv file into an .skx file.

Configuring the Service Component of TIBCO Adapter for COM

- 1. Start TIBCO Designer and open the service instance.



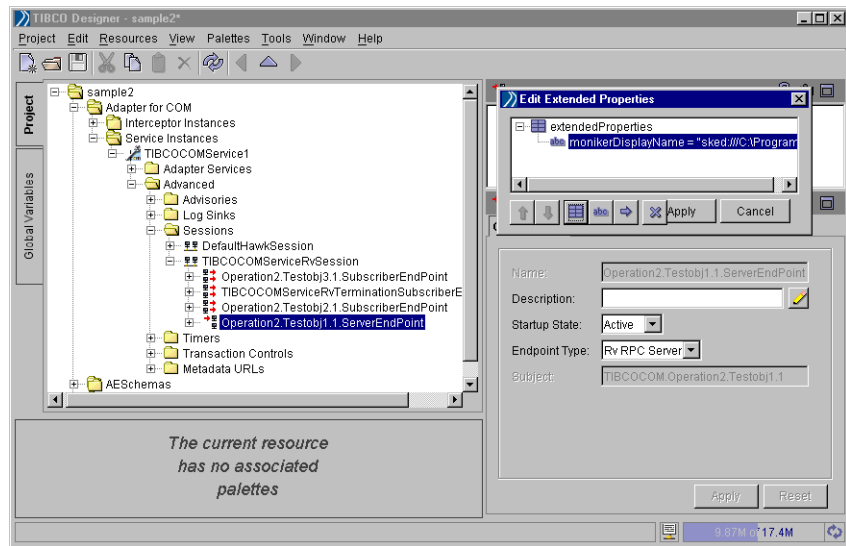
- 2. Click on either a Request Response or a Publication endpoint and then select **Resources > Edit Extended Properties**.



- 3. Add a string (abc) extended property named monikerDisplayName (case sensitive) with the appropriate value for the BizTalk sked moniker, then click **Apply**.

The syntax for the BizTalk sked moniker is:

`monikerDisplayName=sked://<hostname><!groupmanager>/<filepath>/<port name>`



For example, in the screen above, the following string extended property was added:

```
monikerDisplayName=sked:///C:\Program Files\Microsoft BizTalk
Server\Tutorial\Schedule\Lab\Buyer1.skx/Port_1
```

Here *filepath* is the path to the BizTalk .skx file that contains the compiled BizTalk schedule, and *portname* is the name of the incoming port in that schedule. When the service component of the adapter starts, it will use the sked moniker to create a COM interface. Then, when the service component makes a call through a method on that interface, BizTalk will intercept the call, transform the data passed as parameters into a BizTalk message, and hand the message over to the incoming port.

Pass BizTalk Messages to TIBCO Messaging Transport

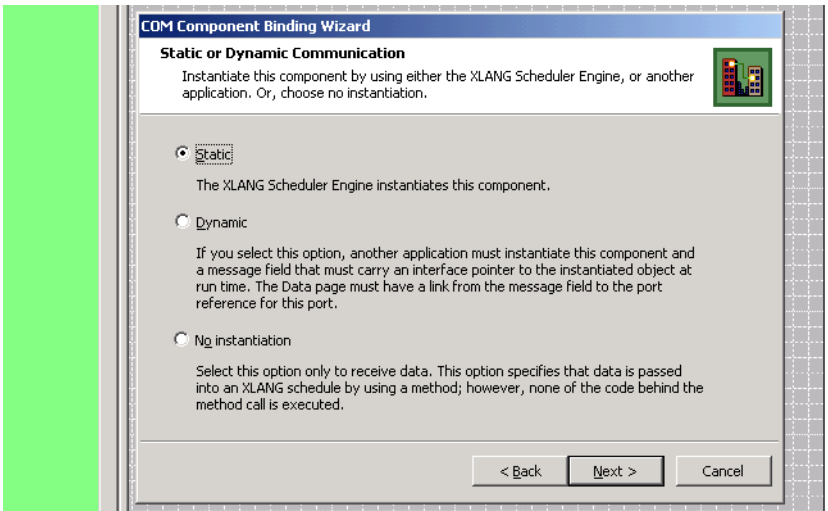
To allow BizTalk messages to pass to TIBCO messaging transport, you must create an outgoing COM component port in BizTalk.

1. With the same .skv file open in BizTalk Orchestration Designer, select the **Business Process** tab.
2. From the **Implementation** palette, drag a **COM component** port into the right-hand panel. This will display the COM Component Binding Wizard.

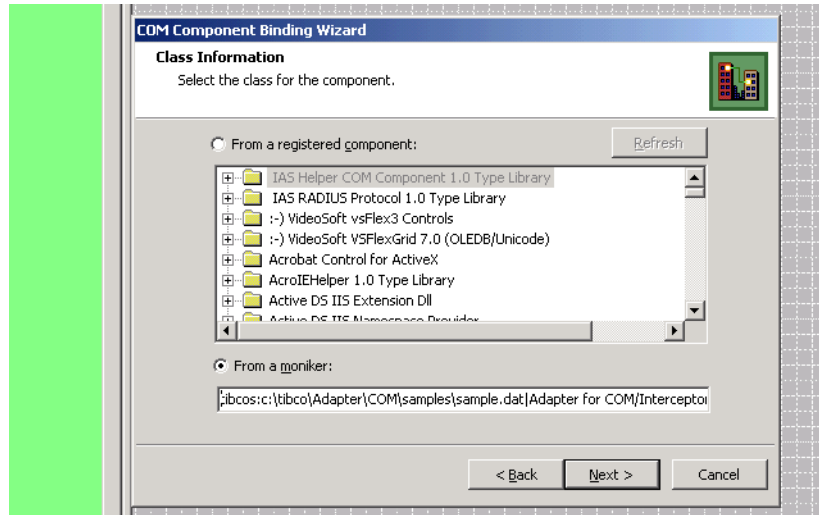
3. Enter a name for the port and click **Next**.



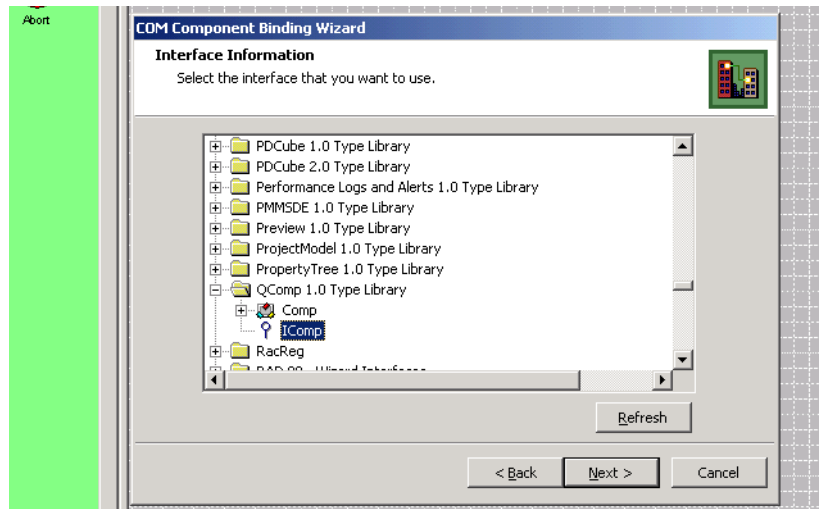
4. On the Static or Dynamic Communication page, select the **Static** radio button and click **Next**.



5. On the Class Information page, select the **From a moniker** radio button, enter a tibco or tibcos moniker, and click **Next**.



6. On the Interface Information page, select the **IComp** interface to associate it with the port. Click **Next**.



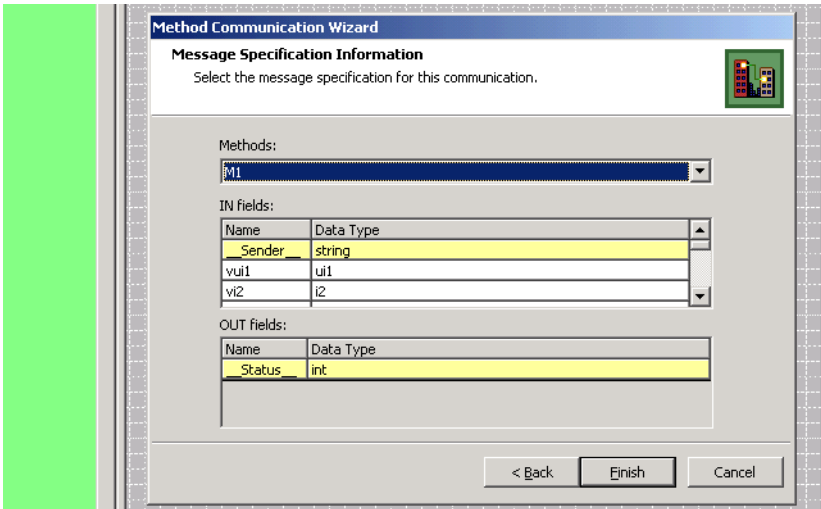
7. On the Advanced Port Properties page, set any advanced properties you want and click **Finish**. The COM Component Binding Wizard closes.
8. Drag the right-hand connector from an **Action** shape to the left-hand connector of the port. The Method Communication Wizard displays.

9. Select the **Initiate a synchronous method call** radio button and click **Next**.



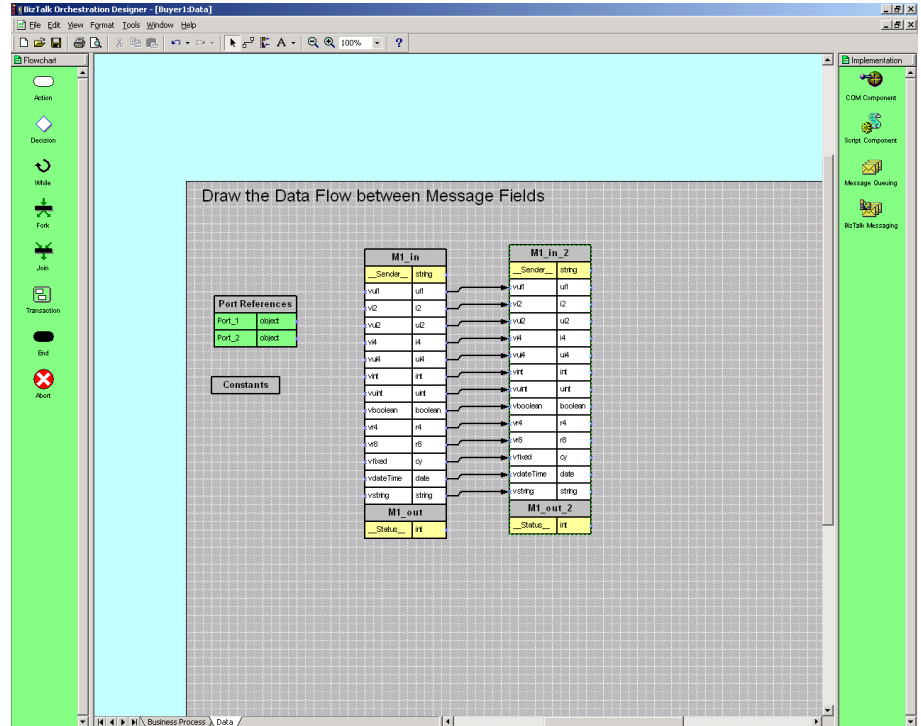
10. Press **Next** to skip the Message Information page.
11. On the Message Specification Information page, select **M1** on the COM interface to define the BizTalk message format and click **Finish**.

The type information from the parameters of the method is used to define the message format. The **Action** shape is now connected to the outgoing port.



12. (Optional) If you select the **Data** tab in the Orchestration Designer, you will see the outgoing message format displayed.

13. Connect the fields in the incoming message to the required fields in the outgoing message.



14. Compile and save the .skv file into a .skx file.



Microsoft BizTalk server 2004(beta) does not support a port implementation directly using a COM component. Therefore, the section on Creating an incoming "COM component" port in BizTalk on page 164 for passing messages to Microsoft BizTalk is not applicable for the BizTalk Server 2004 (beta) release. Due to this limitation, the adapter cannot directly talk to the BizTalk Server 2004 (beta) release.

MSMQ Demonstration Example

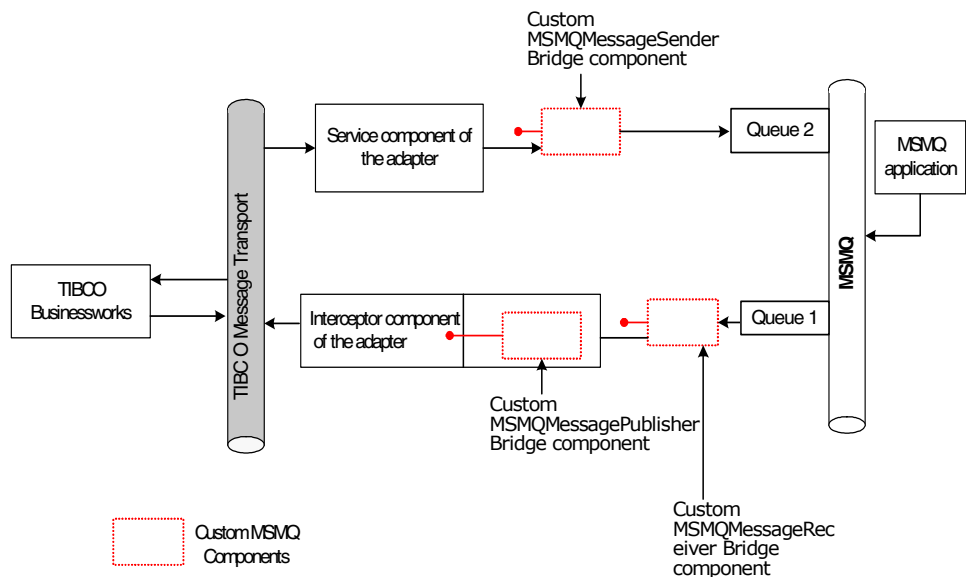
This example discusses how TIBCO ActiveEnterprise applications can be integrated with Microsoft's message queuing product - MSMQ, using TIBCO Adapter for COM. You must be familiar with COM, MSMQ, TIBCO adapters and TIBCO BusinessWorks concepts to be able to successfully run this example. See the MSDN documentation for further details on Message queuing COM components.

The adapter enables seamless integration of TIBCO applications with applications running on the Microsoft platform through the COM interfaces exposed by these applications. The functionality offered by MSMQ can be accessed via its COM interface, and this helps the adapter to implement an integration solution that bridges TIBCO applications such as TIBCO BusinessWorks or other TIBCO Adapters with MSMQ.

Architecture

Figure 6 provides a high-level conceptual overview of the various pieces involved in this example:

Figure 6 Architecture



As seen in the figure, the example setup comprises of the following components:

- TIBCO BusinessWorks that is publishing and subscribing to messages on the TIBCO messaging transport.
- MSMQ Applications that are sending and receiving messages to and from MSMQ.
- The adapter publishing and subscribing to messages on the TIBCO messaging transport.
- The MSMQ Bridge components. These are custom COM components that implement the desired MSMQ message handling. There are essentially three bridge components:

MSMQMessageSender COM component: This COM component is invoked by the service component of the adapter. The service component of the adapter creates an instance of the MSMQMessageSender component, and listens for messages corresponding to this component. When a message is received, the service translates it to the appropriate method call on the MSMQMessageSender component. The MSMQMessageSender component then inspects the parameters passed to the method invocation; creates a corresponding MSMQ message and places it in an MSMQ queue.

MSMQMessageReceiver COM component: This component implements the notification call back interface required by MSMQ. MSMQ invokes a method on this component when a message is placed in an MSMQ queue. When the call back is called by MSMQ, this component uses the MSMQMessagePublisher COM component to publish a corresponding TIBCO Message.

MSMQMessagePublisher component: This component is used by the MSMQMessageReceiver component to publish messages on the TIBCO messaging transport. On receiving an MSMQ message that has been placed in the queue, the MSMQMessageReceiver component will invoke a method on the MSMQMessagePublisher component. This method invocation is intercepted by the interceptor component of the adapter. The interceptor component then marshals the method parameters passed to this call into a message, and sends the message over the TIBCO messaging transport.

Example Scenario

On posting a message to the source queue, the MSMQMessageReceiver component is called by MSMQ. This component, in turn, invokes a method on the MSMQMessagePublisher component. When the method on the MSMQMessagePublisher component is invoked, the interceptor component of the adapter intercepts this invocation and this message is then published on the TIBCO messaging transport.

TIBCO BusinessWorks receives the message from the TIBCO messaging transport, suitably maps the message to be written on to a destination queue and publishes it to the TIBCO messaging transport.

The subscription service of the adapter receives the message from the TIBCO transport and converts it automatically into an invocation of a method on the MSMQMessageSender COM component. The MSMQMessageSender component will then inspect the parameters passed to the method invocation, create a corresponding MSMQ message, and place it in the destination MSMQ queue.

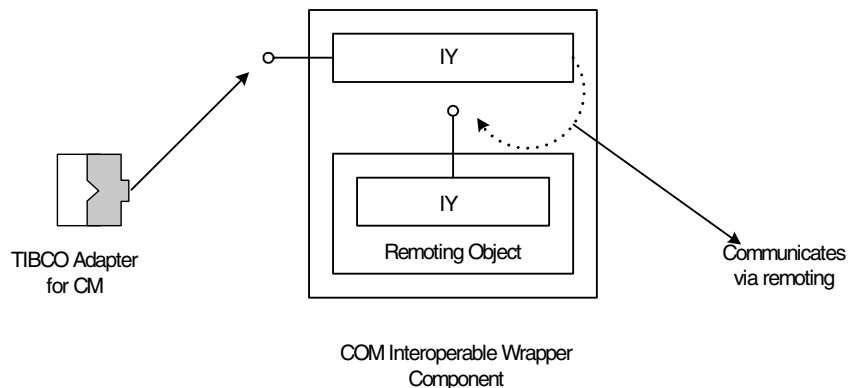
.NET Remoting Examples

Microsoft® .NET remoting provides a framework that allows objects to interact with one another across application domains. The framework provides a number of services, including activation and lifetime support, as well as communication channels responsible for transporting messages to and from remote applications. Formatters are used for encoding and decoding the messages before they are transported by the channel.

As .NET remoting is a feature, specific to the .NET framework, .NET remoting applications cannot directly interoperate with the TIBCO adapter for COM, which is a pure Win32 application. In order to be able to communicate with .NET remoting applications, the adapter needs an interoperable bridge or wrapper, which is a piece of software that accepts commands from the adapter, modifies them, and forwards them to the .NET remoting application.

The following figure explains the interaction between the interoperable bridge component, the adapter and the .NET remoting server.

Figure 7 TIBCO Adapter for COM interoperability with .NET Remoting server



The example setup comprises the following components:

- **Remoting object**: This is the remote object that is exposed to the adapter through the interoperable bridge.
- **TIBCO Adapter for COM** which invokes methods on the interoperable bridge.
- **Interoperable bridge or wrapper component**: This is the COM interoperable wrapper around the remoting server and acts as a client to it.

To create the Interoperable Wrapper:

1. To interoperate with the adapter, the bridge component should implement all the guidelines specified in the section Adapter and .NET Component Interoperability on page 193.
2. As the bridge acts as a client to the remoting object, it should create an object of that type and also register the channels required for communicating with the remoting server. This code is implemented in the constructor of the RemotingWrapper for the examples.
3. The interoperable bridge should implement the methods present in the remote object which are to be exposed to the adapter. The implementation of these methods should involve delegating the calls to the corresponding methods on the .NET remote object.

Examples Scenario

When the adapter invokes methods on the interoperable bridge, the bridge forwards the method calls to the corresponding methods in the remoting server. The .NET remoting system intercepts the calls; forwards them to the remoting server. The results are returned to the interoperable bridge, which is returned to the adapter.

Appendix A Data Types

This appendix describes COM data types that the adapter supports, the mapping of those data types to TIBCO ActiveEnterprise types and fully-scoped names of the data types used in the repository.



For more information on TIBCO ActiveEnterprise data types, refer to the *TIBCO Adapter SDK Programmer's Guide*.

Topics

- *TIBCO Adapter for COM Data Types, page 182*
- *VARIANTs, page 184*
- *SAFEARRAYs, page 185*
- *User Defined Types (UDTs), page 186*
- *General Comments on Data Types, page 190*

TIBCO Adapter for COM Data Types

TIBCO Adapter for COM supports both basic and complex data types.
Table 15 maps COM basic data types to TIBCO ActiveEnterprise data types.

Table 15 COM Data Types and TIBCO ActiveEnterprise Data Types Mapping

COM Data Type	Unscoped TIBCO ActiveEnterprise Data Type Name	Scoped ActiveEnterprise Data Type Name
VT_I1	i1	/tibco/public/scalar/ae/i1
VT_UI1	ui1	/tibco/public/scalar/ae/ui1
VT_I2	i2	/tibco/public/scalar/ae/i2
VT_UI2	ui2	/tibco/public/scalar/ae/ui2
VT_I4	i4	/tibco/public/scalar/ae/i4
VT_UI4	ui4	/tibco/public/scalar/ae/ui4
VT_INT	i4	/tibco/public/scalar/ae/i4
VT_UINT	ui4	/tibco/public/scalar/ae/ui4
VT_BOOL	boolean	/tibco/public/scalar/ae/boolean
VT_R4	r4	/tibco/public/scalar/ae/r4
VT_R8	r8	/tibco/public/scalar/ae/r8
VT_CY	fixed.19.4	/tibco/public/scalar/ae/fixed.19.4
VT_DATE	dateTime	/tibco/public/scalar/ae/dateTime
VT_BSTR	string or char.n ^a	/tibco/public/scalar/ae/string or /tibco/public/scalar/ae/char.n
VT_VOID	void	/tibco/public/scalar/ae/void
VT_PTR	sequence [any]	/tibco/public/sequence/ae/class/ae /sequence [any]

a. *n* represents the maximum number of characters allowed in a string.



TIBCO Adapter SDK (starting with v3.2) supports Unicode characters. This allows applications to use Unicode strings and to send them over TIBCO Rendezvous. TIBCO Adapter for COM leverages this functionality by using the UTF16_LittleEndian encoding support of TIBCO Adapter SDK.

VARIANTS

The supported COM data types for an element contained by a VARIANT are:

- Any of the basic COM data types described above
- UDT
- SAFEARRAY

Thus, if x is one of these supported COM data types and y is the corresponding unscoped TIBCO ActiveEnterprise data type name:

Table 16 *VARIANT Data Types*

COM Data Type	Unscoped TIBCO ActiveEnterprise Data Type Name	Scoped ActiveEnterprise Data Type Name
VT_VARIANT (vt == VT_x at runtime)	any (y at runtime)	/tibco/public/scalar/ae/any

SAFEARRAYs

The supported COM data types for an element of a SAFEARRAY are:

- Any of the basic COM data types described above
- UDT
- VARIANT

Thus, if x is one of these supported COM data types and y is the corresponding unscoped TIBCO ActiveEnterprise data type name:

Table 17 *SAFEARRAY Supported COM Data Types*

COM Data Type	Unscoped TIBCO ActiveEnterprise Data Type Name	Scoped ActiveEnterprise Data Type Name
VT_SAFEARRAY	Sequence<y>	/tibco/public/scalar/ae/sequence<y> or
SAFEARRAY(x) in IDL	MSequence	/tibco/public/class/ae/.../sequence<y>

The scoped data type name depends on whether y is a TIBCO ActiveEnterprise scalar type or a TIBCO ActiveEnterprise class type.



TIBCO Adapter for COM supports only SAFEARRAYs of a single dimension. That is, the ActiveEnterprise class sequence <sequence<x>> is not supported.

Visual Basic runtime makes the declaration and handling of SAFEARRAYs transparent to the application developer. Therefore, inconsistent behavior may occur while handling SAFEARRAY/UDT's as [in, out] parameters from the Visual Basic runtime, depending on how the SAFEARRAY/UDT's are defined. The interceptor component of the adapter returns valid data back to the Visual Basic runtime, irrespective of how the [in, out] parameters are defined in Visual Basic. To ensure consistent behavior from the Visual Basic runtime, you must allocate memory in the client for the [in, out] parameters.

User Defined Types (UDTs)

This section discusses COM and ENUM UDTs.

COM Data Types

The supported COM data types for member variables in a struct are:

- Any of the basic COM data types described above
- UDT
- SAFEARRAY
- VARIANT

A COM User Defined Type (UDT, COM type VT_RECORD) is mapped to a TIBCO ActiveEnterprise class. For example:

Table 18 UDT Supported COM Data Types

COM Data Type	Unscoped TIBCO ActiveEnterprise Data Type Name	Scoped TIBCO ActiveEnterprise Data Type Name
VT_RECORD Order	Order MInstance	/tibco/public/class/ae/.../Order

ENUMs

The user-defined type ENUMs are mapped as i4 in the repository.

Table 19 ENUMs

COM Data Type	Unscoped TIBCO ActiveEnterprise Data Type Name	Scoped ActiveEnterprise Data Type Name
ENUM	i4	/tibco/public/scalar/ae/i4

ADO Recordset Data Object

The adapter accepts recordset pointers as method parameters and does the intrinsic conversion to/from a sequence of ActiveEnterprise classes to be in sync with the TIBCO Adapter SDK data types. However, the adapter does not support any embedded pointers as data types. Only the recordset pointers and ADO connection object pointers are supported for the current release, and not

IDispatch. For example, in Visual Basic, a recordset can be treated as an object in method parameters. However, in the type library, the object is represented as an IDispatch pointer. Such scenarios, where objects are used in place of the recordset, are not supported. The recordset should be explicitly declared as ADODB.Recordset as method parameters for the adapter to work. The Import Metadata option on the Import Schema tab in TIBCO Designer will accept recordset pointers as method parameters, both in and out types. Among the different types of recordset available in the ADO libraries, only the snapshot recordset is supported, which corresponds to ODBC static cursors.

As all data contained in the recordset object essentially needs to be transported over the TIBCO messaging transport by the adapter, the recordset object implicitly or explicitly cannot contain data types that are not supported by the adapter. If the data type of any field contained in the recordset object is different from the data types supported by the adapter, the adapter will display an error. For example, if the recordset gets the data from an underlying SQL table where the data type for a column is binary, it cannot be transported by the adapter over the TIBCO messaging transport. The adapter displays an exception if any data types other than the supported ones are used in the recordset data object.

A list of mapping for data types in different layers is provided:

Table 20 Mapping for Data Types

SQL Server Data Type	Supported?	ADO Equivalent	OLE Equivalent	AE SDK Equivalent
bigint	No	adBigInt	-	-
binary	No	adBinary	-	-
bit	Yes	adBoolean	VT_BOOL	MBool
char	Yes	adChar	VT_BSTR	MStringData
datetime	Yes	adDBTimeStamp	VT_DATE	MDateTime
decimal	No	adNumeric	-	-
float	Yes	adDouble	VT_R8	MReal
image	No	adVarbinary	-	-
int	Yes	adInteger	VT_INT	MInteger
money	Yes	adCurrency	VT_CY	MFixed

Table 20 Mapping for Data Types

SQL Server Data Type	Supported?	ADO Equivalent	OLE Equivalent	AE SDK Equivalent
nchar	Yes	adWChar	VT_BSTR	MStringData
ntext	Yes	adWChar	VT_BSTR	MStringData
numeric	No	adNumeric	-	-
nvarchar	Yes	adWChar	VT_BSTR	MStringData
real	Yes	adSingle	VT_R4	MReal
smalldatetime	Yes	adTimeStamp	VT_DATE	MDateTime
smallint	Yes	adSmallInt	VT_I2	MInteger
smallmoney	Yes	adCurrency	VT_CY	MFixed
sql_variant	No	adVariant	-	-
sysname	Yes	adWChar	VT_BSTR	MStringData
text	Yes	adChar	VT_BSTR	MStringData
timestamp	No	adBinary	-	-
tinyint	No	adVarbinary	-	-
uniqueidentifier	No	adGUID	-	-
varbinary	No	adVarbinary	-	-
varchar	Yes	adChar	VT_BSTR	MStringData

All the data types listed in the table are Microsoft SQL server specific. Other data providers might have slight variations on the data types and the mapping to ADO equivalent might vary depending on the provider.

ADO Connection Object

The adapter accepts ADO Connection Object pointers as method parameters and converts to and from ActiveEnterprise `connclass` classes to stay in sync with the TIBCO Adapter SDK data types. Such objects can only be used by TIBCO ActiveEnterprise clients (like TIBCO BusinessWorks) that want to utilize the COM Server methods using ADO Connection Objects. The TIBCO ActiveEnterprise clients will typically use recordset parameters provided in the other COM Server methods. All the requirements described for recordset parameters in the previous section are applicable. See the examples for more information.

Note that the adapter does not support any embedded pointers as data types; only recordset pointers and ADO Connection pointers are supported for the current release, and not IDispatch. For example, in Visual Basic, an ADO Connection Object can be treated as an object in method parameters. However, in the type library, the object is represented as an IDispatch pointer. Such scenarios, where objects are used in place of the ADO Connection object, are not supported. The ADO Connection object should be explicitly declared as `ADODB.Connection` as method parameters for the adapter to work.

The `Import Metadata` option on the `Import Schema` tab in TIBCO Designer will accept ADO Connection Object pointers as method parameters, both `in` and `out` types. The COM server should provide a method which has the Connection object as `[out]` or `[out, retval]`. Other COM server methods should have the Connection object as the `[in]` parameter. As a best practice, the COM server should also provide a method to release the ADO Connection; this method will also have the ADO Connection object as an `[in]` parameter.

COM Data Type	Unscoped TIBCO ActiveEnterprise Data Type Name	Scoped TIBCO ActiveEnterprise Data Type Name
VT_PTR	connclass	/tibco/public/class/ae/connclass

General Comments on Data Types

The adapter does not support embedded pointers. Therefore, no member variable of a UDT may be a pointer. For example, the following COM UDT (as expressed in IDL) would be illegal except for the ADO Recordset pointer and ADO connection object pointer described earlier:

```
struct xyz
{
    short* m_pShort;
    BSTR* m_pbstr;
    SAFEARRAY(int)* m_pSA;
    struct xyz* m_pxyz;
};
```

Note; however, that the types BSTR and SAFEARRAY, though intrinsically pointers, are allowed. For example:

```
struct xyz
{
    short m_Short;
    BSTR m_bstr;
    SAFEARRAY(int) m_pSA;
    struct xyz m_xyz;
};
```

Furthermore, VT_BYREF is not allowed in VARIANT's. For example, the following VARIANT VARTYPE would be illegal:

```
VT_I2 | VT_BYREF
```

Finally, the element type of a SAFEARRAY may not be a pointer. For example, the following SAFEARRAY would be illegal:

```
SAFEARRAY(int*)
```

The adapter allows top-level pointers only for parameters with the <out> attribute. Thus, the following method signature would be illegal:

```
HRESULT InvalidMethod(<in> int* pint);
```

The best source of information for data types supported by the adapter is the `sample.dat` example shipped with the product. The file `main.cpp` in that project contains numerous examples of supported data types.

Appendix B **Interoperability with Microsoft .NET Components**

This appendix describes the interoperability issues between the adapter and Microsoft .NET components.

Topics

- *COM and .NET Interoperability, page 192*
- *Adapter and .NET Component Interoperability, page 193*
- *Sample Modules, page 195*

COM and .NET Interoperability

.NET allows developers to leverage existing code by providing an interface to their COM components. The .NET framework provides a rich infrastructure for development and backward interoperability with COM. The `System.Runtime.InteropServices` namespace in the .NET framework class library contains interoperability services that allow .NET-managed code to interoperate with COM components.

Calling .NET Components from a COM Client

Although COM clients can call code that is exposed in a public class by .NET components, .NET code is not directly accessible to COM clients. To use .NET code from a COM client, developers need to create a proxy known as a COM callable wrapper (CCW).

There are two prerequisites to creating a .NET class that will be used by COM clients:

- Explicitly define an interface in the .NET code, and have the class implement the interface. Implementing functionality through interfaces provides a major benefit for COM clients. .NET keeps interfaces consistent with previous versions when generating CCWs, which prevents changes to the .NET server from breaking COM clients.
- Any class that is visible to COM clients must be declared public. The tools that create the CCW only define types based on public classes. The same rule applies to methods, properties, and events that will be used by COM clients.

Calling COM Components from a .NET Client

Although .NET clients can call code that is exposed through interfaces by classic COM components, COM code is not directly accessible to .NET clients. To use unmanaged COM code from a managed .NET client, developers need to create a proxy known as a Runtime callable wrapper (RCW). COM metadata information, contained in a type library, needs to be converted to .NET metadata before the managed code can access unmanaged COM code at runtime.

Adapter and .NET Component Interoperability

The `System.Runtime.InteropServices` namespace in the .NET framework class library contains interoperability services that allow .NET-managed code to interoperate with COM components. Specifically, a .NET managed client can call methods on a COM interface through a Runtime Callable Wrapper (RCW). Likewise, a COM client can call methods on a managed interface through a COM Callable Wrapper (CCW). Since the interceptor component of the adapter is just a COM interface, a managed client can call methods on the interceptor through an RCW. Likewise, since the service component of the adapter is just a COM client, it can call methods on any managed interface through a CCW.

.NET and COM Interoperation with the Adapter

The 5.3 version of the adapter supports all data types for .NET interoperability with .NET framework 1.1. If you are developing a .NET component or a .NET client to be used with TIBCO Adapter for COM for interoperation, you must take care of the following:

- The `Register for COM Interop` option should be set to `true` for the .NET components project configuration properties. Any managed component that you develop on your own and that you want to use in conjunction with the adapter must be registered for COM interoperability.
- When the COM type information is generated for the .NET component, Visual Studio .NET does not write the `TypeLib` key (under the `classid` key) to the registry. For successful .NET, COM and adapter interoperability, the `TypeLib` key must be added to the registry. To do this, a custom method marked with the `<ComRegisterFunction>` attribute can be added to the .NET class. The .NET class adds the `TypeLib` key when Visual Studio .NET registers the COM coclass. If the project is created in VB.NET, the `TypeLib` key is automatically generated in the `AssemblyInfo` file. This `TypeLib` key has to be used in the COM register function. To remove the `TypeLib` key from the registry, a custom method marked with the `<ComUnregisterFunction>` can be added to the .NET class. The .NET class removes the `TypeLib` key when Visual Studio .NET unregisters the coclass.
- All the types (UDT's, interfaces, coClasses and so on) should be implemented with the `<GuidAttribute>` attribute. Also, for the coClasses, the `<ClassInterface(ClassInterfaceType.None)>` attribute should be specified. This prevents the language compiler from generating the class interface. This is necessary since the adapter requires the interface implemented by the coclass to be the `<default>` interface.

- .NET clients that use the adapter should call the `Marshal.BindToMoniker()` method available in the `System.Runtime.InteropServices` namespace to conform to the moniker-based activation required for the adapter.

Sample Modules

This section explains how to use the sample .NET component and client that ships with the adapter.

Sample source code, written in C#, is provided in the `\examples\CSharp` directory of the adapter installation, and illustrates how the adapter can interoperate with C# code. Although the sample source code is written in C#, code can be written in any .NET CLS-compliant language (for example, Visual Basic .NET).

The `examples\CSharp\Operation` directory contains a C# class library project that defines a class `Testobj1`. The `Testobj1` class is similar to the C++ `Testobj1` class in the `examples\Operation\Operation2` directory of the adapter installation.

The `examples\CSharp\TestInterceptor` directory contains a C# console application project that defines a client program. This `TestInterceptor` C# client is similar to the C++ client in the `examples\testinterceptor` directory of the adapter installation.

The `SampleCSharp.dat` available in the `examples` directory is a sample repository that contains:

- The metadata for the `Testobj1` class
- A Request Response Invocation Service (client) endpoint, `Operation.Testobj1.CSharpClient` in the `TIBCOCOMInterceptor1` adapter instance, that references the `Testobj1` class
- A Request Response (server) endpoint, `Operation.Testobj1.CSharpServer` in the `TIBCOCOMService1` adapter instance, that references the `Testobj1` class.

To use the sample module

4. Build the `Operation` project using Visual Studio .NET (VS .NET). Modify the path for the assembly key file in `AssemblyInfo.cs` to match the adapter installation.

When you build the project, VS .NET will register the `Testobj1` class for COM interoperability, generating a COM type library, `Operation.tlb`, in the `tibco\adapter\adcom\5.3\examples\CSharp\Operation\bin\Debug` directory, and will write most of the required entries into the Microsoft Windows registry.

Before building the project, you can verify that VS .NET will register `Testobj1` for COM interoperability. In the Solution Explorer window,

right-click the **Operation** node and select **Properties** from the shortcut menu; select **Configuration Properties > Build > Outputs** and make sure **Register for COM Interop** is set to **True**.



There is one required key that VS.NET does not write to the registry. This is the `TypeLib` key for the `Testobj1` coclass. For successful .NET, COM, and adapter interoperability, the `TypeLib` key must be added to registry. The `Testobj1` class has a method marked with the `<ComRegisterFunction>` attribute that adds the `TypeLib` key when VS .NET registers the COM coclass and a method marked with the `<ComUnregisterFunction>` that removes the `TypeLib` key when VS .NET unregisters the coclass. These methods are run automatically when VS.NET builds the project and you do not need to do anything for the `Testobj1` class. However, any managed component that you develop on your own and that you want to use in conjunction with the COM and .NET interoperability, and the adapter must have this `TypeLib` key added. You can either use your own methods marked with the `<ComRegisterFunction>` and `<ComUnregisterFunction>` attributes, as illustrated in the `Testobj1` class, or you can add the `TypeLib` key manually. The GUID's for the type library and the coclass are defined by `GuidAttribute` attributes (in the `Assembly.cs` and `Class1.cs` respectively).

5. The `SampleCSharp.dat` already contains the type information from the `Operation.tlb` type library, but you must reimport this information after building the `Operation` project. To do so, import the type library from the `Import Schema` tab in TIBCO Designer. The `Import Schema` tab is available when you select the `Advanced Services` folder in the project panel:



You must run the metadata import utility to import type information for any managed component you develop.

6. Build the `TestInterceptor` project using Visual Studio .NET.
7. Start the Service component of the adapter:

```
TIBCOCOMService -system:repourl
<Install_Dir>\adapter\adcom\5.3\Samples\SampleCSharp.dat -system:configurl "Adapter
for COM/Service Instances/TIBCOCOMService1"
```

8. Run the `TestInterceptor` executable.

Appendix C **Monitoring the Adapter Using TIBCO Hawk**

This chapter explains how to use TIBCO Hawk microagents to monitor and manage the adapter.

Topics

- *Overview, page 198*
- *Starting TIBCO Hawk Software, page 199*
- *The Auto-Discovery Process, page 200*
- *Invoking Microagent Methods, page 201*
- *Available Microagents, page 204*

Overview

TIBCO Hawk is a sophisticated tool for enterprise-wide monitoring and managing of all distributed applications and systems. System administrators can use it to monitor adapters in a wide area network of any size. TIBCO Hawk can be configured to monitor system and adapter parameters and to take actions when predefined conditions occur. These actions include: sending alarms that are graphically displayed in the TIBCO Hawk display, sending email, paging, running executables, or modifying the behavior of a managed adapter.

Unlike other monitoring applications, TIBCO Hawk relies on a purely distributed intelligent agent architecture using publish or subscribe to distribute alerts. TIBCO Hawk uses TIBCO Rendezvous for all messaging and thus gains the benefits and scalability from the TIBCO Rendezvous features of publish/subscribe, subject name addressing, interest-based routing, and reliable multicast.

TIBCO Hawk is a purely event-based system that uses alerts. The agents are configured with rules that instruct them on everything from what and how to monitor to what actions to take when problems are discovered. Thus the workload is fully distributed throughout the enterprise. Every agent is autonomous in that it does not depend on other components to perform its functions.

The TIBCO Hawk Enterprise Monitor consists of these components:

- **Display**—GUI front end that displays alarms and provides editors to create rule bases, create tests, view messages, and invoke microagents to request information or initiate an action.
- **Agents**—Intelligent processes that perform monitoring and take actions as defined in rules.
- **Rulebases**—Rules that are loaded by agents to determine agent behavior.
- **Application Management Interface (AMI)**—Manages network applications via TIBCO Rendezvous and supports communication between a network application and monitoring TIBCO Hawk agents, including the ability to examine application variables, invoke methods, and monitor system performance.
- **Microagents**—Feed information back to TIBCO Hawk and expose action methods to rulebases.

For more information, see the TIBCO Hawk documentation.

Starting TIBCO Hawk Software

The TIBCO Hawk agent can be configured to start automatically during the system boot cycle. See the *TIBCO Hawk Installation and Configuration* guide for information about starting TIBCO Hawk.

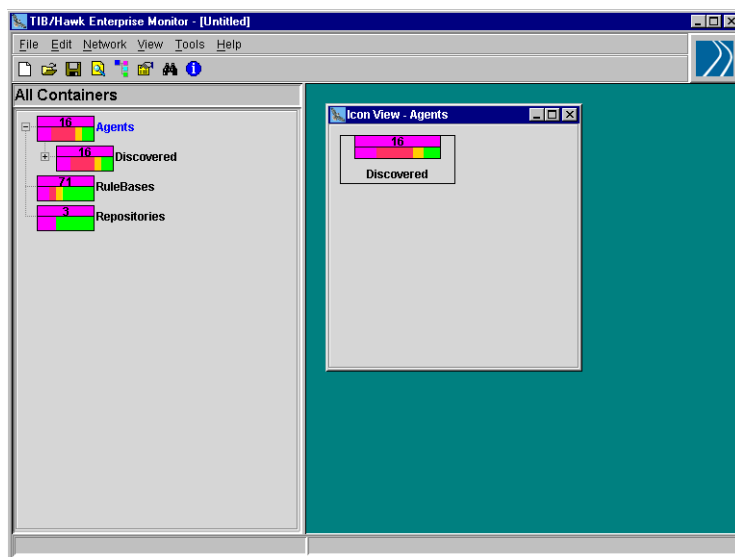
The *TIBCO Hawk Administrator's Guide* explains how to start the TIBCO Hawk Display.

The guides are included in your TIBCO Hawk software installation area.

The Auto-Discovery Process

After you start an instance of TIBCO Hawk Display, it continually discovers machines running TIBCO Hawk Agents on your network. Container icons are created for each agent, and arranged hierarchically in clusters. By default, agent icons are clustered according to subnets.

At first, the Agents container is empty. Its counter displays a value of zero and, on the right, the Discovered counter is also at zero. Both icons are initially green in color to show that no alerts, or warning messages, are in effect. As agents are discovered, the counters increment to reflect the current number of discovered agents:



Monitored network nodes are arranged in a hierarchical tree of containers. Clicking a container in the left panel displays nested items on the right.

Icon colors change to reflect the highest level of alert found on discovered agents. For explanations of icon elements and characteristics, see your *TIBCO Hawk Administrator's Guide*.

Invoking Microagent Methods

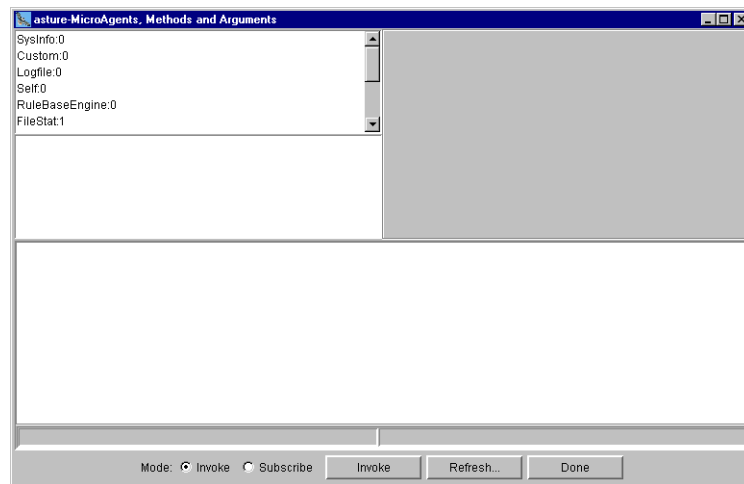
A set of default microagents is loaded when a TIBCO Hawk Agent is started. When you install and start the adapter, its microagents are dynamically added to the local agent.

To invoke a microagent method:

1. Start TIBCO Hawk Display, then right-click on the agent icon and select **Get Microagents**.

If TIBCO Hawk security is implemented on your system and you do not have access to microagents on this agent, an error dialog displays. Select another agent, or contact your system administrator to obtain access.

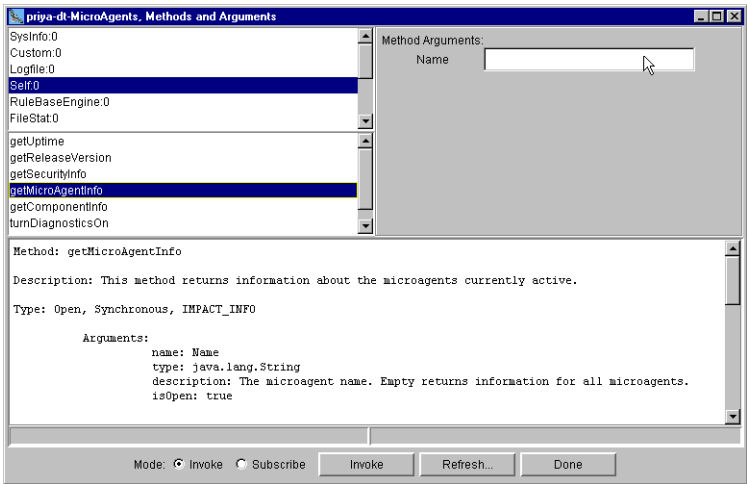
The Microagents, Methods and Arguments dialog displays. The panel on the upper left lists microagents you can access on the current agent.



This dialog has two modes, Invoke and Subscribe. Invoking a method immediately returns a single set of current results. Subscribing provides updates of current results at regular intervals. Radio buttons at the bottom of the dialog control these modes.

2. Click a microagent name, such as **Self**, to display a list of associated methods and text descriptions in the panels below.

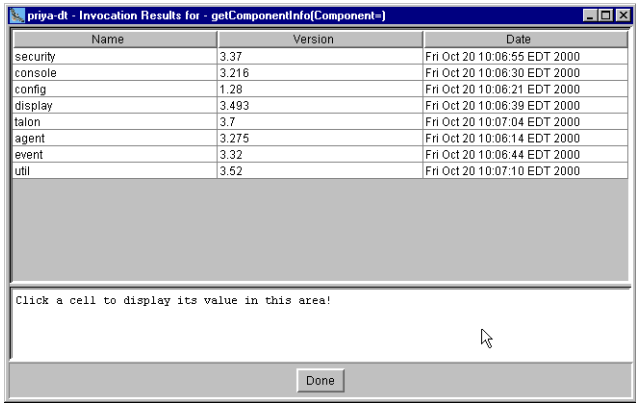
- 3. Click the name of the method to invoke, such as `getComponentInfo`.



If the method accepts arguments, fields for each argument display in the upper right panel. Detailed help text displays in the lower panel.

- 4. Specify any arguments for the method invocation.
- 5. Verify that the **Invoke** radio button is selected.
- 6. Click the **Invoke** button to invoke the selected method.

The Invocation Results dialog displays the results returned by the method.



- 7. Click **Done** to close the dialog.

These steps describe how to interactively invoke a microagent method and receive a single set of results in TIBCO Hawk Display. You can also use a microagent method as the data source of a TIBCO Hawk rule. Rules automatically receive method results, apply tests to evaluate them, then take action if necessary. For more information on building TIBCO Hawk rules and rule bases, see your *TIBCO Hawk Administrator's Guide*.

Available Microagents

Each adapter has two microagents, a standard TIBCO Hawk microagent named `COM.TIBCO.ADAPTER.xyz` where `xyz` is the adapter configuration name and a custom microagent. The microagents provide:

- Queries that return information about the state of the adapter. This can be an important tool for seeing the internals of an adapter and debugging it if something appears wrong. For example, methods can return information about threads, internal queues, or connections to the target system. Using these methods, one might be able to identify certain bottlenecks or gauge how successfully an adapter is scaling with respect to the current environment.
- Updates of the adapter runtime parameters. This includes retrieving the current runtime parameters and setting new runtime parameters without restarting the adapter. An example of this is getting and setting the polling interval. Updating a runtime parameter through the Hawk microagent only affects the setting of the instance that is running. It does not make a permanent change of the setting in either the repository or the `.tra` file.

By default both the standard and custom microagents are available at runtime. You can disallow adding custom methods to the standard microagent when deploying the adapter by changing the `addCustomHawkMethodstoStdMAgent` property value in the adapter's property file. See [Adapter Properties File](#) on page 100 for more information.

The following table lists each method available for the adapter and page on which the method is explained.

Table 21 *Microagent Methods*

Method	Description	Page
Standard Microagent Methods		
<code>activateTraceRole()</code>	Activates a mapping of a role to a sink at runtime.	207
<code>deactivateTraceRole()</code>	Deactivates a mapping of a roles to sinks at runtime.	208
<code>getAdapterServiceInformation()</code>	Returns information about the services implemented by this adapter.	209
<code>getComponents()</code>	Returns information about the publisher, subscriber and <code>IODescriptor</code> .	210

Table 21 Microagent Methods (Cont'd)

Method (Cont'd)	Description (Cont'd)	Page
<code>getConfig()</code>	Returns basic configuration information. More specific information is accessed by the more specific methods.	211
<code>getConfigProperties()</code>	Returns all attributes and elements for the given repository object.	212
<code>getHostInformation()</code>	Returns standard and extended application information.	213
<code>getRvConfig()</code>	Returns information about all TIBCO Rendezvous sessions defined.	214
<code>getStatus()</code>	Returns general status information, such as the number of TIBCO Rendezvous messages received and published, the number of errors since the last call, the PID of the application, and more.	215
<code>getTraceSinks()</code>	Returns information about sinks to which traces currently go.	216
<code>_onUnsolicitedMsg()</code>	Returns the configuration ID, application name, version, and date for this adapter instance.	217
<code>_onUnsolicitedMsg()</code>	Displays alert messages sent to the current adapter.	217
<code>preRegisterListener()</code>	Preregisters an anticipated listener.	218
<code>reviewLedger()</code>	Returns information retrieved from the ledger file of a certified messaging session for a publisher adapter.	219
<code>setTraceSinks()</code>	Adds a role or changes the file limit of a previously specified sink.	221
<code>stopApplicationInstance()</code>	Stops the running adapter instance.	222
<code>unRegisterListener()</code>	Unregisters a currently preregistered listener.	223

Table 21 *Microagent Methods (Cont'd)*

Method (Cont'd)	Description (Cont'd)	Page
Custom Microagent Methods		
<code>getAdapterServiceInformation()</code>	Returns the total number of objects processed for all the schemas.	209
<code>getActivityStatisticsBySchema()</code>	Returns the total number of objects processed for the given schema by each service that uses the schema.	225
<code>getActivityStatisticsByService()</code>	Returns information about the services implemented by this adapter.	226
<code>getQueueStatistics()</code>	Returns the current count of elements in any internal queue used by the adapter.	227
<code>resetActivityStatistics()</code>	Resets all the counts for the activity statistics.	229
<code>resetThreadStatistics()</code>	Resets all the counts for the thread statistics.	230

activateTraceRole()

Activates a mapping of a role to a sink at runtime. This replaces the now-deprecated `setTraceSink()` TIBCO Hawk method.

Input Parameters	Type	Description
Role Name	string	Name of the role to activate.
Sink Name	string	Name of the sink for which to activate the role.

deactivateTraceRole()

Deactivates a mapping of a roles to sinks at runtime.

Input Parameters	Type	Description
Role Name	string	Name of the role to activate.
Sink Name	string	Name of the sink for which to activate the role.

getAdapterServiceInformation()

Returns information about the services implemented by this adapter.

Input Parameter	Type	Description
Service Name	string	Name of the service from which to get information. Default is ALL.

Returns	Type	Description
Line	integer	Sequential row number.
Service Name	string	Name of the service as defined at design-time.
Endpoint Name	string	Name of the endpoint used for this service.
Type	string	Type of the endpoint, for example, publisher or subscriber.
Quality of Service	string	Quality of service for the endpoint. For example RVCM or JMS Persistent.
Subject	string	Subject defined for this endpoint.
Class	string	Class associated with the endpoint.
Number of Messages	integer	Number of messages processed for this endpoint.

getComponents()

Returns information about the currently active TIBCO Hawk components such as publishers, subscribers, or timers.

Input Parameters	Type	Description
Component Name	string	Name of the component. If no value is enter, all components display.
Component Type	string	Any of Publisher, Subscriber, Timer, or IODescriptor. The default value is All.

Returns	Type	Description
Instance ID	string	Name of this adapter instance as defined at design-time.
Adapter Name	string	Name of the adapter.
Component Name	string	Name of the component.
Component Type	string	The name of the TIBCO Adapter SDK class for this component, such as Publisher, Subscriber, or IODescriptorSource. For more information about the class, see your TIBCO Adapter SDK documentation.
Session Name	string	Name of the session.
Description	string	Information about this component, for example, time interval, signal type, and validating the publisher or subscriber.

getConfig()

Retrieves generic configuration information. More specific configuration information is accessed through separate methods.

Returns	Type	Description
Instance ID	string	Configuration ID of this adapter.
Adapter Name	string	Name of the adapter.
Repository Connection	string	URL of the repository used for adapter instance.
Configuration URL	string	Location of the adapter project; either a file name or configuration URL.
Command	string	Command line arguments used to start the adapter.

getConfigProperties()

Returns all attributes and elements for the given repository object.

Input Parameter	Type	Description
Property	string	Name of the property for which elements (tags) and attributes are desired. For example, agentone/startup. If no value is given, all properties are returned.

Returns	Type	Description
Element Name	string	Repository directory for the property.
Attribute Name	string	Name of the repository object attribute.
Attribute Value	string	Value of the repository object attribute.
Line	integer	Line number in which this property is defined in the project file.

getHostInformation()

Return standard and extended application information set. It returns the following information.

Returns	Type	Description
Name	string	Name of the property.
Value	string	Value of the property.

getRvConfig()

Returns information about the TIBCO Rendezvous session defined by this adapter. Information about all currently defined sessions is returned if no `sessionName` is provided.

Input Parameter	Type	Description
Session Name	string	Name of the TIBCO Rendezvous session for which configuration is required. If not given, information about all sessions is returned. The default is all.

Returns	Type	Description
Instance ID	string	Configuration ID of this adapter.
Adapter Name	string	Name of the adapter.
Session Name	string	Name of the session.
Service	string	Service parameter for this session.
Daemon	string	Daemon parameter for this session.
Network	string	Network parameter for this session.
Synchronous?	boolean	Returns 1 if this is a synchronous session, 0 otherwise.
Session Type	string	Type of session; one of M_RV, M_RVCM, or M_RVCMQ.
Certified Name	string	Name of this certified session.
Ledger File	string	Ledger file for this certified messaging session. Returns the empty string for sessions that are not certified messaging sessions.
CM Timeout	string	Timeout for this certified messaging session. Returns the empty string for sessions that are not certified messaging sessions.

getStatus()

Retrieves basic status information about the adapter.

This information is fairly limited; for more detail, additional methods are provided (*getConfig()* on page 75 and *getRvConfig()* on page 77).

Returns	Type	Description
Instance ID	string	Configuration ID for this adapter instance.
Adapter Name	string	Name of the adapter.
Uptime	integer	Number of seconds since startup.
Messages Received	integer	Number of TIBCO Rendezvous messages received.
Messages Sent	integer	Number of TIBCO Rendezvous messages published.
New Errors	integer	Number of errors since the last call to this method.
Total Errors	integer	Total number of errors since startup.
Process ID	integer	Process ID of the application.
Host	string	Name of host machine on which this adapter is running.

getTraceSinks()

Returns information about sinks to which traces currently go.

Input Parameters	Type	Description
Sink Name	string	Name of the sink for which you need information. If no name is specified, information about all sinks is returned. Default is all.
Role Name	string	Name of the role for which you need information for the specified sink or sinks. Default is all.

Returns	Type	Description
Instance ID	string	Name of this adapter instance as a string.
Adapter Name	string	Name of the application for this sink.
Sink Name	string	Name of the sink
Sink Type	string	Type of this sink. One of fileSink, rvSink, hawkSink, stderrSink.
Roles	string	Roles this sink supports, as a string. For example warning, error, debug.

_onUnsolicitedMsg()

Displays all alert messages sent from the adapter or an error if not successful.

preRegisterListener()

Preregister an anticipated subscription service. Some sending applications can anticipate requests for certified delivery even before the listening applications start running. In such situations, the publication service can preregister subscription services, so TIBCO Rendezvous software begins storing outbound messages in the publication service ledger. If the listening correspondent requires old messages, it receives the backlogged messages when it requests certified deliver.

Input Parameters	Type	Description
Session Name	string	Name of the session that anticipates the listener.
Publisher Name	string	Name of the component for which the listener should be preregistered.
Listener Session Name	string	Name of the subscription service to preregister.

Returns OK if the subscription service was preregistered successfully, false otherwise.

reviewLedger()

Returns information retrieved from the ledger file of a TIBCO Rendezvous certified messaging session.

Before invoking this method, ensure that the certified messaging publisher adapter has established a certified delivery agreement with its subscriber agents.

Input Parameters	Type	Description
Session Name	string	Name of the TIBCO Rendezvous session for which ledger information is desired (default is all).
Subject	string	Name of the subject for which ledger information is desired.

Returns	Type	Description
Session Name	string	Name of the TIBCO Rendezvous CM session to which this information applies.
Subject	string	Subject name for this session.
Last Sent Message	integer	Sequence number of the most recently sent message with this subject name.
Total Messages	string	Total number of pending messages with this subject name.
Total Size	integer	Total storage (in bytes) occupied by all pending messages with this subject name. If the ledger contains ten messages with this subject name, then this field sums the storage space over all of them.
Listener Session Name	string	Within each listener submessage, the Listener Session Name field contains the name of the delivery-tracking listener session.

Returns (Cont'd)	Type	Description
Last Confirmed	string	Within each listener submessage, the Last Confirmed field contains the sequence number of the last message for which this listener session confirmed delivery.
Line	integer	Row number in ledger file.
Unacknowledged Messages	integer	Number of RVCN messages pending for this listener. The value is computed by subtracting the last sent sequence number from the last acknowledged sequence number.

setTraceSinks()

Adds a role or changes the file limit of a previously specified sink.

Input Parameters	Type	Description
Sink Name	string	Name of the sink for which you want to add a role or change the file limit.
Role Name	string	Name of the role you want to add to this sink (warning, error, debug, or user defined). Default is all.
File Size	integer	Maximum file size for this sink. This parameter is ignored if the sink specified by sinkName is not a file sink.

Returns OK if successful or an error if not successful.

stopApplicationInstance()

Stops the specified adapter by calling the internal `stop()` method. This method returns OK if successful or an error if not successful.

unRegisterListener()

Unregister a currently preregistered subscription service.

Input Parameters	Type	Description
Session Name	string	Name of the session that anticipates the subscription service.
Publisher Name	string	Name of the publication service to which the subscription service is preregistered.
Listener Session Name	string	Name of the subscription service to unregister.

This method returns true if the subscription service was unregistered successfully, false otherwise.

getActivityStatistics()

Returns the total number of objects processed for all the schemas, based on the request type. Also, returns the number of success and error objects.

Input Parameter	Type	Description
GetSubTotalBy	string	Indicates how to group the subtotals, by Service or Operation.

Returns	Type	Description
Name	string	Service name or All Services which represents the final tally of all the services
Total	integer	Total number of objects processed including both success and failures.
Success	integer	Total number of objects successfully processed.
Failure	integer	Total number of objects that caused an error during processing.
MeasurementInterval	integer	Displays the time (in seconds) since last time the adapter was reset, or if never reset, since the adapter started.

getActivityStatisticsBySchema()

Returns the total number of objects processed for the given schema by each service that uses the schema. Also, returns the number of success and error objects.

Parameter	Type	Description
Schema Name	string	Name of the schema.

Returns	Type	Description
Service Name	string	Name of the service that is associated with the specified schema.
Total	string	Total number of objects processed for this schema for a publication service. Total number of objects received for this schema for a subscription service.
Success	string	Number of objects that were successfully identified for this schema, which will be published or written to a file.
Failure	string	Number of objects that were identified for this schema but were not published because the header of the schema failed validation for a publication service, or was written to a file because the schema was not associated with a subscriber for a subscription service.

getActivityStatisticsByService()

Returns statistics about the data handled by a given adapter service or all adapter services since the time the adapter was started.

Input parameter	Type	Description
Service Name	string	Name of service to get the statistics for. If no service name is given, performance statistics for all services is returned.

Returns	Type	Description
Service Name	string	Service name
Schema Name	string	Name of top level schema processed by this service.
Operation	string	Type of operation this service provides.
Total	integer	Total number of objects processed, both success and failures.
Success	integer	Total number of objects successfully processed.
Failure	integer	Total number of objects that caused an error during processing.
MeasurementInterval	integer	Displays the time (in seconds) since last time the adapter was reset, or if never reset, since the adapter started.
LineIndex	string	Concatenated string of Service Name and Operation separated by a comma.

getQueueStatistics()

Return the current count of elements in any internal queue used by the adapter. This includes the TIBCO Rendezvous event queues automatically spawned by Rendezvous for each adapter.

Returns	Type	Description
QueueID	string	Unique identification of a particular queue.
QueueType	string	Type or key that will match this queue to a thread or connection.
QueueCount	integer	Current number of elements in the queue.
MaxQueueSize	integer	Maximum number of elements in the queue.
MeasurementInterval	integer	Displays the time (in seconds) since last time the adapter was reset, or if never reset, since the adapter started.

getThreadStatistics()

Return the operation counts of the current threads.

Returns	Type	Description
ThreadID	string	Unique identification of a particular thread.
ThreadType	string	Type that tells what part of the adapter this thread belongs. Valid types include "Publisher", "Subscriber", "RPC", or "Connection".
TaskType	string	One-word description of the tasks this thread processes.
TaskCount	integer	Number of tasks processed by this thread.
MeasurementInterval	integer	Displays the time (in seconds) since last time the adapter was reset, or if never reset, since the adapter started.

resetActivityStatistics()

Resets all the counts for the activity statistics.

resetThreadStatistics()

Resets all the counts for the thread statistics.

Appendix D **Trace Messages**

This appendix explains the trace messages that are logged to a location specified at configuration time.

Topics

- *Overview, page 232*
- *Trace Message Fields, page 234*
- *Status Messages, page 237*

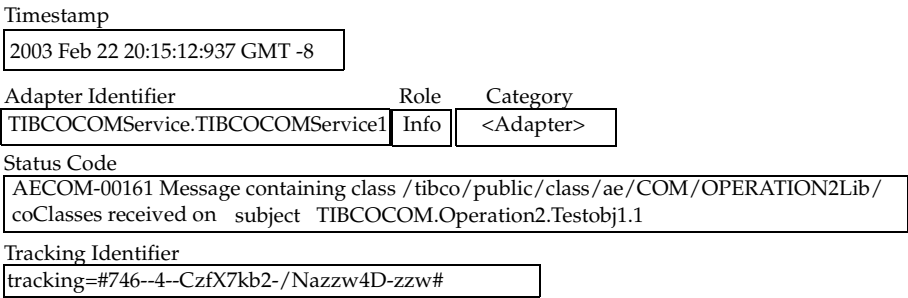
Overview

Trace messages provide information about adapter activities. The messages are logged to the console where the runtime adapter was started and to a log file. Trace messages can also be redirected to the TIBCO Hawk Display application, or sent to other applications using the TIBCO Rendezvous transport.

Each trace message can include the following fields:

<Timestamp> <Adapter Identifier> <Role> <Category> <Status Code>
<Tracking Identifier>

The above fields are explained in Trace Message Fields on page 234. The following diagram shows an example trace message and calls out the fields.



Example Trace Messages

The following trace messages were written during a session where TIBCO Adapter for COM received an object from an application, then processed the object.

The first message indicates that TIBCO Adapter for COM has started. The timestamp indicates when the adapter started, and the role indicates that the trace message is informational, which means the activity is normal for the adapter. The category is identified, and the corresponding status code is displayed. The status code indicates that the adapter started successfully.

2003 Feb 22 20:14:51:718 GMT -8 COMadapter.COMAdapterConfiguration
Info <Configuration> AECOM-00161 TIBCO Adapter for COM successfully
initialized

The next set of trace messages indicates the adapter received an object that was sent on the TIBCO Rendezvous subject, TIBCOCOM.Operation2.Testobj1.1. The #746--4--CzfX7kb2-/Nazzw4D-zzw# tracking identifier included in the trace message uniquely identifies the message. The application from which the message originated provided the identifier.

```

2001 Nov 20 15:11:36:078 GMT -8 TIBCOCOMService.TIBCOCOMService1
Info <Adapter> AECOM-00161 Received request
(pid-320|tid-908|TIBCOCOMService1|Operation2.Testobj1.1.Server|/tibco/public/class/ae/COM/OPERATION2Lib/interfaces/ITestobj1|TestAcrossTypeLibs--pid-1596|tid-2608|TIBCOCOMInterceptor1|Operation2.Testobj1.1.Client|/tibco/public/class/ae/COM/OPERATION2Lib/coClasses/Operation2.Testobj1.1|TestAcrossTypeLibs).
tracking=#746--4--CzfX7kb2-/Nazzw4D-zzw#

```

Trace Message Fields

Each trace message includes the following fields:

Table 22 Tracing Fields

Field Name	Description
Timestamp	Timestamp of occurrence. For example, 2003 Feb 22 20:14:51:718 GMT -8.
Adapter Identifier	Name of the adapter that wrote the trace message. This is a combination of the adapter acronym and adapter configuration name. For example, the application identifier, ADB.publisher1 identifies a TIBCO Adapter for ActiveDatabase service named publisher1.
Role	<div>A role can be:<ul style="list-style-type: none">• Info. Indicates normal adapter operation. No action is necessary. A tracing message tagged with Info indicates that a significant processing step was reached and has been logged for tracking or auditing purposes. Only info messages preceding a tracking identifier are considered significant steps.• Warn. An abnormal condition was found. Processing will continue, but special attention from an administrator is recommended.• Error. An unrecoverable error occurred. Depending on the error severity, the adapter may continue with the next operation or may stop altogether.• Debug. A developer-defined tracing message. In normal operating conditions, debug messages should not display.<p>When configuring the adapter you define what roles should or should not be logged. For example, you may decide not to log Info roles to increase performance.</p></div>

Table 22 Tracing Fields

Field Name	Description
Category	<p>One of the following:</p> <ul style="list-style-type: none"> • Adapter. The adapter is processing an event. • Configuration. The adapter is reading configuration information. • Metadata. The adapter is retrieving metadata from the COM Server type library. • Palette. The adapter is interacting with the palette. • Publisher Service. The publication service is reporting this trace message. • Request-Response Client Service. The request-response invocation service is reporting this trace message. • Request-Response Server Service. The request-response service is reporting this trace message. • Shutdown. The adapter is shutting down. • Startup. The adapter is starting. • Subscription Service. The subscription service is reporting this trace message. • System. This category is not linked to a specific event process. The trace message may be related to a Microsoft Windows service related messages, memory allocation, file system error, and so on. • TibRvComm. The adapter is communicating with TIBCO Rendezvous. • XML. The adapter is parsing XML documents.
Status Code	<p>Unique code for the message and description. Status codes are identified by a unique number and description. If a trace message includes an error or warn role, the status code documentation includes a resolution. See Status Messages on page 237 for details.</p>
Tracking Identifier	<p>A unique identifier that is "stamped" on each message by the originating adapter. The tracking identifier remains in effect from a message's beginning to its completion as it is exchanged by TIBCO applications. If the adapter is the termination point of the message, the tracking identifier is not displayed in the trace message.</p> <p>You cannot modify the tracking identifier format or configure what information is displayed.</p>

Table 22 Tracing Fields

Field Name	Description
Application Information	Target COM Server information added to the tracking info to trace the message back to its source. Set initially by the originating adapter and carried forward. It is augmented by each intermediate component.

Status Messages

Status Code	Role	Category	Resolution
AECOM-00002	<trace_message>		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00003	<error_message>		
	errorRole	Adapter	Check for the previous error messages.
AECOM-00004	Cannot convert GUID into string.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.
AECOM-00005	Cannot convert wide character string to multi byte character string.		
	errorRole	Adapter	There was a problem with the system. Restart the process.
AECOM-00006	Cannot convert multi byte character string to wide character string.		
	errorRole	Adapter	There is a problem with the system. Restart the process.
AECOM-00007	Cannot construct _guid_t object.		
	errorRole	Adapter	Internal error. Register the COM component again in the system registry.
AECOM-00008	Cannot generate GUID.		
	errorRole	Adapter	Unable to get the Ethernet or token-ring hardware address of the system. Check with the manufacturer.

Status Code	Role	Category	Resolution
AECOM-00009	Invalid GUID <GUID> () <HRESULT>.		
	errorRole	Adapter	Internal error. Restart the process.
AECOM-00010	Cannot open registry key.		
	errorRole	Adapter	Check the permissions for the current Microsoft Windows user.
AECOM-00011	Cannot query registry value.		
	errorRole	Adapter	Check the permissions for the current Microsoft Windows user.
AECOM-00012	Cannot determine LIBID for CLSID <CLSID>.		
	errorRole	Adapter	Register the COM component again in the Microsoft Windows registry.
AECOM-00013	Cannot open LIBID key <LIBID> in registry () <HRESULT>.		
	errorRole	Adapter	Register the COM component again in the Microsoft Windows registry.
AECOM-00014	Cannot get latest type library version for LIBID key <LIBID>.		
	errorRole	Adapter	Register the COM component again in the Microsoft Windows registry.
AECOM-00015	Cannot load type library for LIBID key <LIBID> () <HRESULT>.		
	errorRole	Adapter	Make sure that the registered DLL is in its physical location.
AECOM-00016	Cannot load type library for CLSID <CLSID> () <HRESULT>.		
	errorRole	Adapter	Make sure that the registered DLL is in its physical location.
AECOM-00017	Cannot get type information for CLSID <CLSID> () <HRESULT> ()<error_info>.		
	errorRole	Adapter	Check the permissions for the current Microsoft Windows user.

Status Code	Role	Category	Resolution
AECOM-00018	Cannot get type information for interface <Interface> on coclass <coClass> () <HRESULT>.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.
AECOM-00019	Invalid number of methods on interface <Interface> () <HRESULT>.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.
AECOM-00020	Simple type encountered where complex type (UDT or SAFEARRAY) is required ()<Type>		
	errorRole	Adapter	In the repository, use complex type (UDT or SAFEARRAY) instead of simple type.
AECOM-00021	Cannot get type information attributes for CLSID <CLSID>.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.
AECOM-00022	Call to ITypeInfo::GetRefTypeInfo failed () <HRESULT>.		
	errorRole	Adapter	Make sure that the registered DLL is in its physical location.
AECOM-00023	Call to ITypeInfo::GetContainingTypeLib failed () <HRESULT>.		
	errorRole	Adapter	Make sure that the registered DLL is in its physical location.
AECOM-00024	Call to ITypeInfo::GetDocumentation failed () <HRESULT>.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.

Status Code	Role	Category	Resolution
AECOM-00025	Call to ITypeInfo::GetVarDesc failed () <HRESULT>.		
	errorRole	Adapter	Internal error while reading the type library. Make sure that the correct DLL is registered.
AECOM-00026	Call to ITypeLib::FindName failed <HRESULT>, <error_info>.		
	errorRole	Adapter	Internal error while reading the type library. Make sure that the correct DLL is registered.
AECOM-00027	Call to ITypeLib::GetDocumentation failed () <HRESULT>.		
	errorRole	Adapter	Internal error while reading the type library. Make sure that the correct DLL is registered.
AECOM-00028	Call to ITypeLib::FindName returned <name> ITypeInfo interfaces for name () <name>.		
	errorRole	Adapter	Internal error while reading the type library. Make sure that the correct DLL is registered.
AECOM-00029	Invalid VARTYPE () <vartype>.		
	errorRole	Adapter	Parameter type used is not supported in this version of the adapter. Use only the supported parameter types.
AECOM-00030	Invalid type name () <type_name>.		
	errorRole	Adapter	This version of the adapter does not support the type name used. Use only the supported type names.
AECOM-00031	Cannot create record for UDT <UDTname>.		
	errorRole	Adapter	Internal error. Restart the process.
AECOM-00032	Cannot destroy record for UDT <UDTname>.		
	errorRole	Adapter	Internal error. Restart the process.

Status Code	Role	Category	Resolution
AECOM-00033	The number of member variables in a COM UDT exceeds the number of fields in an AE class () <class>.		
	errorRole	Adapter	Error message not used.
AECOM-00034	Cannot get AE class description () <class>.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.
AECOM-00035	Operations mismatch for the class <class>. The following operations are present in the COM type library, but not in the AE repository: <class>. The following operations are present in the AE repository, but not in the COM type library: <type_library>.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.
AECOM-00036	Parameters mismatch for operation <operation>. The following parameters are present in the COM type library, but not in the AE repository: <repository>. The following operations are present in the AE repository, but not in the COM type library: <type_library>.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.
AECOM-00037	Direction mismatch for parameter <parameter> in operation <operation>.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.
AECOM-00038	Return type mismatch for operation <Type>.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.

Status Code	Role	Category	Resolution
AECOM-00039	Parameter type mismatch for parameter <parameter> in operation <operation>.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.
AECOM-00040	No AE class associated with endpoint <endpoint> () <HRESULT>.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.
AECOM-00041	The default interface on the COM coclass associated with the AE class <class> (endpoint <endpoint>) has more than one method.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.
AECOM-00042	Method <method> of the default interface on the COM coclass associated with the AE class <class> (endpoint <endpoint>) has more than one parameter.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.
AECOM-00043	Param <parameter> of method <method> of the default interface on the COM coclass associated with the AE class <class> (endpoint <endpoint>) has invalid direction.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.
AECOM-00044	Param <parameter> of method <method> of the default interface on the COM coclass associated with the AE class <class> (endpoint <endpoint>) has invalid type.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.

Status Code	Role	Category	Resolution
AECOM-00045	The MSubscriber <subscriber> received a message containing an unsupported complex type (<Type>).		
	errorRole	Adapter	This version of the adapter does not support the type used. Use only the supported types.
AECOM-00046	The MSubscriber <subscriber> received a message containing a non-matching complex type (<Type>).		
	errorRole	Adapter	This version of the adapter does not support the type used. Use only the supported types.
AECOM-00047	Operation <operation> is missing the TIBCOCOMADAPTER_EXCEPTION of type "string".		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.
AECOM-00048	Invalid number of dimensions for SAFEARRAY (<diemension>).		
	errorRole	Adapter	Restart the process.
AECOM-00049	Invalid element type (<Type>) in SAFEARRAY. Should be <Type>.		
	errorRole	Adapter	Restart the process.
AECOM-00050	Call to SafeArrayAccessData failed (<HRESULT>).		
	errorRole	Adapter	Restart the process.
AECOM-00051	Call to SafeArrayUnaccessData failed (<HRESULT>).		
	errorRole	Adapter	Restart the process.
AECOM-00052	Call to SafeArrayCreate or SafeArrayCreateEx failed (<HRESULT>).		
	errorRole	Adapter	Restart the process.

Status Code	Role	Category	Resolution
AECOM-00053	Call to SafeArrayDestroy failed (<HRESULT>).		
	errorRole	Adapter	Restart the process.
AECOM-00054	Call to SafeArrayGetVartype failed (<HRESULT>).		
	errorRole	Adapter	Restart the process.
AECOM-00055	Call to SafeArrayGetRecordInfo failed (<HRESULT>).		
	errorRole	Adapter	Restart the process.
AECOM-00056	A SAFEARRAY contains an invalid element type (<Type>).		
	errorRole	Adapter	Element type used is not supported. Use only the supported element types, details of which can be found in the user's guide.
AECOM-00057	A SAFEARRAY cannot be the element type of a SAFEARRAY (<Type>).		
	errorRole	Adapter	Element type used is not supported. Use only the supported element types, details of which can be found in the user's guide.
AECOM-00058	A VARIANT contains an invalid type (<Type>).		
	errorRole	Adapter	Element type used is not supported. Use only the supported element types, details of which can be found in the user's guide.
AECOM-00059	Cannot initialize a type description for a type contained by a VARIANT (<Type>). Potential memory leak.		
	errorRole	Adapter	Problem with the System. Restart the process.
AECOM-00060	Call to IRecordInfo::GetName failed (<HRESULT>).		
	errorRole	Adapter	Make sure that the registered DLL is in its physical location.

Status Code	Role	Category	Resolution
AECOM-00061	Call to GetRecordInfoFromTypeInfo failed (<HRESULT>).		
	errorRole	Adapter	Problem with the System. Restart the process.
AECOM-00062	Call to IRecordInfo::GetTypeInfo failed (<HRESULT>).		
	errorRole	Adapter	Make sure that the registered DLL is in its physical location.
AECOM-00063	Cannot create thread initialized event.		
	errorRole	Adapter	Restart the process.
AECOM-00064	Cannot create thread.		
	errorRole	Adapter	Restart the process.
AECOM-00065	Thread exited for unknown reason.		
	errorRole	Adapter	Restart the process.
AECOM-00066	Cannot start dispatching Rendezvous events.		
	errorRole	Adapter	Restart the process.
AECOM-00067	Pool of event dispatching threads has not been created.		
	errorRole	Adapter	Problem with activating the thread(s). Restart the process and system (if possible).
AECOM-00068	Number of event dispatching threads must be greater than 0.		
	errorRole	Adapter	Number of event dispatching threads must be greater than 0. Re-configure the repository option in the repository.
AECOM-00069	Cannot create event dispatching thread.		
	errorRole	Adapter	Problem with creating the thread. Restart the process and system (if possible).

Status Code	Role	Category	Resolution
AECOM-00070	Wait for event dispatching thread to exit timed out (<HRESULT>).		
	errorRole	Adapter	Problem in destroying the threads. Kill the process (if it is still running).
AECOM-00071	MOperationException thrown: <description>.		
	errorRole	Adapter	Error in getting the data from Subscribersreply.Restarttheprocess.'
AECOM-00072	MException thrown: <description>.		
	errorRole	Adapter	Check error description in the event viewer for more information.
AECOM-00073	Unknown exception thrown.		
	errorRole	Adapter	Restart the adapter.
AECOM-00074	<information>		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00075	TIBCOCOMService <service_name> starting with Repo URL ""<repoURL> and Config URL ""<configURL>.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00076	TIBCOCOMService running.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00077	TIBCOCOMService stopping.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00078	TIBCOCOMService initializing.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.

Status Code	Role	Category	Resolution
AECOM-00079	TIBCOCOMService done initializing.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00080	Cannot initialize MTrace logging.		
	errorRole	Adapter	Error message not used.
AECOM-00081	Command line option specified more than once: <option>.		
	errorRole	Adapter	Remove the duplicate command line parameters while executing TIBCOCOMService.
AECOM-00082	Invalid command line option: <option>.		
	errorRole	Adapter	Pass valid command line parameters to TIBCOCOMService.
AECOM-00083	Missing data for command line option: <option>.		
	errorRole	Adapter	Pass valid command line parameters to TIBCOCOMService.
AECOM-00084	Invalid combination of command line options.		
	errorRole	Adapter	Pass valid command line parameters to TIBCOCOMService.
AECOM-00085	Repository URL not specified.		
	errorRole	Adapter	Pass valid repository file name as command line parameter to TIBCOCOMService.
AECOM-00086	Config URL not specified.		
	errorRole	Adapter	Pass a valid Config URL as command line parameter.
AECOM-00087	Programmatic identifier not specified.		
	errorRole	Adapter	Pass a valid ProgID while creating the object.

Status Code	Role	Category	Resolution
AECOM-00088	Cannot cannot create tibco moniker (<HRESULT>).		
	errorRole	Adapter	Check TIBCOMInterceptorService executable file in the system path.
AECOM-00089	IUnknown::QueryInterface failed (<Interface> - <HRESULT>).		
	errorRole	Adapter	Restart adcomInterceptorService.
AECOM-00090	Error calling StartServiceCtrlDispatcher (<error>).		
	errorRole	Adapter	Starting Service failed. Make sure that the service is not already running.
AECOM-00091	Error calling RegisterServiceCtrlHandler.		
	errorRole	Adapter	Service callback registration failed. Restart the service.
AECOM-00092	Error calling SetServiceStatus.		
	errorRole	Adapter	Check the permissions for the current Microsoft Windows user.
AECOM-00093	Cannot open service control manager (<error>).		
	errorRole	Adapter	There can be several possibilities, You may not have access to service control manager.
AECOM-00094	Cannot create NT Service (<error>).		
	errorRole	Adapter	There can be several possibilities, either service already exists or you may not have access to this service.
AECOM-00095	Cannot open NT Service (<error>).		
	errorRole	Adapter	Either service is not installed or You may not have access to service.
AECOM-00096	Cannot delete NT Service.		
	errorRole	Adapter	Deleting the service failed. Try deleting the service again.

Status Code	Role	Category	Resolution
AECOM-00097	Call to CoInitializeEx failed.		
	errorRole	Adapter	Some Microsoft Windows COM Libraries might be missing. Try to reinstall them on the system.
AECOM-00098	Cannot create instance of coclass <class>, interface <Interface> (HRESULT: <HRESULT>).		
	errorRole	Adapter	Make sure that the registered DLL is in its physical location.
AECOM-00099	Unknown exception thrown by implementation of coclass <class> (interface <Interface>, method <method>).		
	errorRole	Adapter	Make sure that the registered DLL is in its physical location.
AECOM-00100	Cannot convert DATE to RVDATE.		
	errorRole	Adapter	Error in converting DATE to RVDATE format. Use the date formats supported by RV.
AECOM-00101	Cannot convert RVDATE to DATE.		
	errorRole	Adapter	Error message not used.
AECOM-00102	Cannot convert CY to BSTR.		
	errorRole	Adapter	Problem in converting the currency value. Check the validity of the currency value.
AECOM-00103	Cannot convert BSTR to CY.		
	errorRole	Adapter	Problem in converting the string to currency value. Check the validity of the currency value.
AECOM-00104	Invalid data type.		
	errorRole	Adapter	Error message not used.

Status Code	Role	Category	Resolution
AECOM-00105	Cannot access termination subscriber (<subscriber>).		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00106	Cannot add termination event listener.		
	errorRole	Adapter	Problem with the termination subscriber in the repository. Please reconfigure it correctly.
AECOM-00107	Cannot create stop event for TIBCOCOMService.		
	errorRole	Adapter	Restart adcomService.
AECOM-00108	Wait for TIBCOCOMService stop event failed.		
	errorRole	Adapter	Restart adcomService.
AECOM-00109	Call to SetEvent failed (<error>).		
	errorRole	Adapter	Restart adcomService.
AECOM-00110	No rv</cm/tx>RpcServers specified in repository.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00111	No rv</cm/tx>Subscribers specified in repository.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00112	Downcast of MData to MGlobalNameData failed (<endpoint>).		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.

Status Code	Role	Category	Resolution
AECOM-00113	Cannot get extended properties for (<extended_property>).		
	errorRole	Adapter	Extended properties are missing in the repository. Please add a configuration option "/class/extendedProperties" in the repository along with correct values. See the user's guide for details.
AECOM-00114	Cannot get value for property (<property>).		
	errorRole	Adapter	Set correct value to "progID" property in the repository.
AECOM-00115	Cannot derive CLSID from Programmatic Identifier (<PROGID>).		
	errorRole	Adapter	Set correct value to "progID" property in the repository.
AECOM-00116	Interface not supported for an AE Class (<PROGID> - <Interface>).		
	errorRole	Adapter	Set correct value to "defaultInterface" property in the repository.
AECOM-00117	Cannot add listener to MSubscriber (<subscriber>).		
	errorRole	Adapter	Problem in adding a subscriber listener. Reconfigure the subscriber correctly in the repository using TIBCO Designer.
AECOM-00118	Call to GetComponentByName failed (<endpoint>).		
	errorRole	Adapter	Endpoint is not configured correctly in the repository. Reconfigure the endpoint in the repository using TIBCO Designer.
AECOM-00119	Downcast of MComponent to MRPCServer failed (<endpoint>).		
	errorRole	Adapter	Cannot create RPC server. Reconfigure the RPC Server correctly in the repository.

Status Code	Role	Category	Resolution
AECOM-00120	Downcast of MComponent to MSubscriber failed (<endpoint>).		
	errorRole	Adapter	Cannot create the Subscriber. Reconfigure the Subscriber correctly in the repository.
AECOM-00121	Downcast of MComponent to MPublisher failed (<endpoint>).		
	errorRole	Adapter	Cannot create the publisher. Reconfigure the Publisher correctly in the repository.
AECOM-00122	Downcast of MData to MInstance failed.		
	errorRole	Adapter	Error message not used.
AECOM-00123	Downcast of MData to MSequence failed.		
	errorRole	Adapter	Error message not used.
AECOM-00124	The value of the parameter <parameter> in the request is NULL.		
	errorRole	Adapter	Error message not used.
AECOM-00125	The value of the parameter <parameter> in the reply is NULL.		
	errorRole	Adapter	Error message not used.
AECOM-00126	Event of type other than MDataEvent received by subscriber (<subscriber>).		
	errorRole	Adapter	Error message not used.
AECOM-00127	Cannot create MInstance from event received by MSubscriber <subscriber> (<description>).		
	errorRole	Adapter	Problem with the Subscriber. Reconfigure the subscriber correctly in the repository.
AECOM-00128	Cannot serialize MInstance.		
	errorRole	Adapter	Problem with the Publisher schema. Reconfigure the schema in repository.

Status Code	Role	Category	Resolution
AECOM-00129	TIBCOCOMInterceptor starting with Repo URL "<repoURL>" and Config URL "<configURL>".		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00130	No rv</cm/tx>RpcClients specified in repository.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00131	No rv</cm/tx>Publishers specified in repository.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00132	Cannot add shutdown notification listener.		
	errorRole	Adapter	Recheck the Shutdown event listener in the repository.
AECOM-00133	Cannot enable shutdown notification (<message>).		
	errorRole	Adapter	Recheck Shutdown event in the repository.
AECOM-00134	Cannot initialize RV</cm/tx> RPC Clients (<description>).		
	errorRole	Adapter	Recheck the endpoint of RPC Clients in the repository.
AECOM-00135	Cannot initialize RV</cm/tx> Publishers (<description>).		
	errorRole	Adapter	Recheck the Publisher endpoints in the repository.
AECOM-00136	Cannot enable MTrace logging.		
	errorRole	Adapter	Error message not used.
AECOM-00137	Cannot enable Cerr logging.		
	errorRole	Adapter	Error message not used.

Status Code	Role	Category	Resolution
AECOM-00138	Cannot enable Event Log logging.		
	errorRole	Adapter	Check the permissions for the current Microsoft Windows user.
AECOM-00139	Cannot get repository information for CLSID (<CLSID>).		
	errorRole	Adapter	Error message not used.
AECOM-00140	AE Repository not currently available.		
	errorRole	Adapter	Error message not used.
AECOM-00141	coclass (<class>) not registered in AE Repository.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.
AECOM-00142	Cannot allocate index for thread local storage (<error>).		
	errorRole	Adapter	Error message not used.
AECOM-00143	Cannot set thread local storage value (<error>).		
	errorRole	Adapter	Error message not used.
AECOM-00144	Cannot get thread local storage value (<error>).		
	errorRole	Adapter	Error message not used.
AECOM-00145	Invalid interface pointer.		
	errorRole	Adapter	Make sure that the registered DLL is in its physical location.
AECOM-00146	No operations defined for class <class>.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.

Status Code	Role	Category	Resolution
AECOM-00147	No parameters defined for operation <class>::<operation>.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.
AECOM-00148	Unsuccessful HRESULT returned by implementation (<HRESULT>).		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.
AECOM-00149	Error getting string from MData (<class>: <description>).		
	errorRole	Adapter	Restart the process.
AECOM-00150	Missing IRecordInfo for record in VARIANT.		
	errorRole	Adapter	Error message not used.
AECOM-00151	Attributes mismatch for the class <class>. The following attributes are present in the COM type library, but not in the AE repository: <AEAttribute>. The following attributes are present in the AE repository, but not in the COM type library: <COMAttribute>.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.
AECOM-00152	The type of the attribute <attribute> in the class <class> is not the same in the AE repository and the COM type library.		
	errorRole	Adapter	Re-import the DLL into the repository using the metadata import utility.
AECOM-00153	Unsupported ActiveEnterprise type (<AEClass>: <class>).		
	errorRole	Adapter	AE Class name used in the repository is not supported in this version of the adapter. Use only the supported AE Class names in the repository. See the user's guide for details.

Status Code	Role	Category	Resolution
AECOM-00154	Successfully created NT Service (<servicename>).		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00155	Successfully deleted NT Service (<servicename>).		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00156	Call to ITypeInfo::GetTypeAttr failed (<HRESULT>).		
	errorRole	Adapter	Make sure that the registered DLL is in its physical location.
AECOM-00157	Cannot get type info for UDT.		
	errorRole	Adapter	Make sure that the registered DLL is in its physical location.
AECOM-00158	Cannot get attribute description <attribute> for AE class <class>.		
	errorRole	Adapter	Make sure that the registered DLL is in its physical location.
AECOM-00159	Invalid instance lifetime value (<value>) in property <property>.		
	errorRole	Adapter	Set the valid value for the "lifetime" in the repository.
AECOM-00160	Sending request (<request>).		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00161	Received request (<request>).		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00162	Sending reply (<reply>).		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.

Status Code	Role	Category	Resolution
AECOM-00163	Received reply (<reply>).		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00164	Publishing document (<document>).		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00165	Received document (<document>).		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00166	Advisory: <advisory> : <message>.		
	errorRole	Adapter	Check the actual error message for details.
AECOM-00167	Advisory: <advisory> : <message>.		
	warnRole	Adapter	
AECOM-00168	TIBCOCOMInterceptor service starting.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00169	TIBCOCOMInterceptor service running.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00170	TIBCOCOMInterceptor service stopping.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00171	Bad service request.		
	errorRole	Adapter	Problem with the Interceptor service. Please reinstall it.

Status Code	Role	Category	Resolution
AECOM-00172	<description>.		
	errorRole	Adapter	Error message not used.
AECOM-00173	Cannot load resource module.		
	errorRole	Adapter	Make sure that the registered DLL is in its physical location.
AECOM-00174	Initialized <endpoint>: Name: <name>; Session: <session>; Subject: <subject>; <information>; Associated class: <class>.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00175	Initializing <endpoint>.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00176	Starting <number> event dispatching thread</s>.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00177	Invalid event of type "<Type>" received by subscriber <subscriber>.		
	errorRole	Adapter	Error message not used.
AECOM-00178	MExceptionEvent received by subscriber <subscriber>: <exception>.		
	errorRole	Adapter	Problem in receiving the data. Check for validity of the incoming data from Interceptor or from external source.
AECOM-00179	Invalid value assigned to messageConfirmBehavior parameter (<messageConfirmBehaviour>).		
	errorRole	Adapter	Please set the valid values for "/extendedProperties/messageConfirmBehavior" configuration option in the Repository.

Status Code	Role	Category	Resolution
AECOM-00180	Downcast of MComponent to MRPCClient failed (<RPCClient>).		
	errorRole	Adapter	Reconfigure the RPC client using TIBCO Designer.
AECOM-00181	Advisory: <advisory> : <description>.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00182	Creation of Recordset object failed.		
	errorRole	Adapter	Verify that the appropriate ADO libraries are present in the system.
AECOM-00183	Call to SafeArrayPutElement failed (<HRESULT>).		
	errorRole	Adapter	Restart the process.
AECOM-00184	Adding field to Recordset object failed (<HRESULT>).		
	errorRole	Adapter	Restart the process.
AECOM-00185	Cannot open the Recordset object created (<HRESULT>).		
	errorRole	Adapter	Verify that the appropriate ADO libraries are present in the system.
AECOM-00186	Cannot populate Recordset object with data received (<HRESULT>).		
	errorRole	Adapter	Verify that the appropriate ADO libraries are present in the system.
AECOM-00187	Call to Recordset object method AddNew failed (<HRESULT>).		
	errorRole	Adapter	Restart the process.
AECOM-00188	Call to Recordset object method MoveFirst failed (<HRESULT>).		
	errorRole	Adapter	Restart the process.

Status Code	Role	Category	Resolution
AECOM-00189	Conversion of Recordset pointer failed.		
	errorRole	Adapter	Verify that the appropriate ADO libraries are present in the system.
AECOM-00190	Recordset object data access failed (<HRESULT>).		
	errorRole	Adapter	Restart the process.
AECOM-00191	Call to Recordset object method MoveNext failed (<HRESULT>).		
	errorRole	Adapter	Restart the process.
AECOM-00192	Cannot close the Recordset object (<HRESULT>).		
	errorRole	Adapter	Restart the process.
AECOM-00193	DataTypeEnum (<datatype>) not supported as a Recordset field.		
	errorRole	Adapter	Element type used is not supported. Use only the supported element types, details of which can be found in the user's guide.
AECOM-00194	Unsupported AE data type used as a Recordset field.		
	errorRole	Adapter	Data type used is not supported. Use only the supported element types, details of which can be found in the user's guide.
AECOM-00195	Publication error: Error occurred while trying to publish. The exception is: <exception>.		
	errorRole	Adapter	Restart the process.
AECOM-00197	No JMSRpcServers specified in repository.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00198	No JMSSubscribers specified in repository.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.

Status Code	Role	Category	Resolution
AECOM-00199	No JMSRpcClients specified in repository.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00200	No JMSPublishers specified in repository.		
	infoRole	Adapter	Indicates normal adapter information. No action necessary.
AECOM-00201	Cannot initialize JMS RPC Clients (<message>).		
	errorRole	Adapter	Recheck the endpoint of JMS RPC Clients in the repository.
AECOM-00202	Cannot initialize JMS Publishers (<message>).		
	errorRole	Adapter	Recheck the endpoint of JMS Publishers in the repository.
AEADCOM_910003	Startup Error. The command-line parameters: <option> has not been specified		
	errorRole	Adapter	Please specify the command-line parameter. For details, see the User's Guide.
AEADCOM_910004	Startup Error. SDK Exception <exception> occurred in the adapter initialization while creating the MAppProperties object. The Repository URL is <repoURL> and the Configuration URL is <configURL>.		
	errorRole	Adapter	Please refer to SDK documentation for Repository URL and Configuration URL specification.
AEADCOM_910005	Startup Error. SDK Error <error> received on starting the adapter after initialization. The Repository URL is <repoURL> and the Configuration URL is <configURL>.		
	errorRole	Adapter	Please verify your repository settings. For details, see the User's Guide..

Status Code	Role	Category	Resolution
AEADCOM_910006	Startup Error. SDK Exception <error> occurred while creating a shutdown listener with parameters <paramter>. The Repository URL is <repoURL> and the Configuration URL is <configURL>.		
	errorRole	Adapter	Please verify your repository settings for validity of configuration for the shut down listener. For details, see the User's Guide.
AEADCOM_910007	Startup Error. Unable to create a connection with the target application <application> using connection parameters <parameter> and the target application error is <error>		
	errorRole	Adapter	Please verify your repository settings for validity of connection parameters. For details, see the User's Guide.
AEADCOM_910008	Startup Error. Received target application error <error> while creating the connection object with the target application <application>. The connection parameters are <parameters> and the connection pool size is <size>.		
	errorRole	Adapter	Please verify your repository settings for validity of connection parameters. For details, see the User's Guide.
AEADCOM_910009	Startup Error. Unable to find target application library/libraries <libraries> in the classpath <classpath> for the target application <application>		
	errorRole	Adapter	Please verify that the client libraries are included in the classpath. For details, see the User's Guide.
AEADCOM_910010	Startup Error. Failed to load properties file containing trace messages from path<path> for Adapter Instance <instance>. Error Message <message>		
	errorRole	Adapter	Please verify that the locale is set to en_US on your machine.

Status Code	Role	Category	Resolution
AEADCOM_910011	Startup Error. Duplicate instance detected with name = <name>. The Repository URL is <repoURL> and the Configuration URL is <configURL>.		
	errorRole	Adapter	Please verify your repository settings for duplicate adapter instance name. For details, see the User's Guide.
AEADCOM_910012	Startup Error. Unable to create a Custom Hawk Micro Agent Named <name> used for <use>.		
	errorRole	Adapter	
AE_ADCOM_START_UP_100	Startup Error. Error calling StartServiceCtrlDispatcher.		
	errorRole	Adapter	Starting Service failed. Make sure that the service is not already running.
AE_ADCOM_START_UP_101	Startup Error. Error calling RegisterServiceCtrlHandler.		
	errorRole	Adapter	Service callback registration failed. Restart the service.
AE_ADCOM_START_UP_102	Startup Error. Error calling SetServiceStatus.		
	errorRole	Adapter	Check the permissions for the current Microsoft Windows user.
AE_ADCOM_START_UP_103	Startup Error. Cannot open service control manager.		
	errorRole	Adapter	There can be several possibilities, You may not have access to service control manager.
AE_ADCOM_START_UP_104	Startup Error. Cannot open NT Service.		
	errorRole	Adapter	Either service is not installed or You may not have access to service.
AE_ADCOM_START_UP_105	Startup Error. Call to CoInitializeEx failed.		
	errorRole	Adapter	Some Microsoft Windows COM Libraries might be missing. Try to reinstall them on the system.

Status Code	Role	Category	Resolution
AE_ADCOM_START_UP_106	Startup Error. Cannot enable Event Log logging.		
	errorRole	Adapter	Check the permissions for the current Microsoft Windows user.
AE_ADCOM_START_UP_107	Number of event dispatching threads must be greater than 0.		
	errorRole	Adapter	Number of event dispatching threads must be greater than 0. Re-configure the repository option in the repository.
AEADCOM_920001	Subscription error. Subscription service <subscription service> listening on <port> received an unexpected event of type = <Type>, Expects event <Type>. The Repository URL is <repoURL> and the Configuration URL is <configURL>		
	errorRole	Adapter	Check the configuration of the application that is publishing the event and make sure that it matches the inbound event definition for the above subscription service. For details, see the User’s Guide.
AEADCOM_920002	Subscription error. Subscription service <subscription service> failed to deserialize the event received on subject <subject> and SDK exception thrown is <exception>. The Repository URL is <repoURL> and the Configuration URL is <configURL>.		
	errorRole	Adapter	Check the configuration of the application that is publishing the event and make sure that it matches the inbound event definition for the above subscription service. For details, see the User’s Guide.

Status Code	Role	Category	Resolution
AEADCOM_920003	Subscription error. Subscription service <subscription service> listening on subject <subject> received inbound event with null data. The Repository URL is <repoURL> and the Configuration URL is <configURL>		
	errorRole	Adapter	Check the configuration of the application that is publishing the event and make sure that it matches the inbound event definition for the above subscription service. For details, see the User's Guide.
AEADCOM_920004	Subscription error. Subscription service <subscription service> listening on subject <subject> could not deserialize the inbound event to MbusinessDocument <document>. The Repository URL is <repoURL> and the Configuration URL is <configURL>		
	errorRole	Adapter	Check the configuration of the application that is publishing the event and make sure that it matches the inbound event definition for the above subscription service. For details, see the User's Guide.
AEADCOM_920005	Subscription error. Subscription service <subscription service> listening on subject <subject>,could not find the tracking data. The Repository URL is <repoURL> and the Configuration URL is <configURL>		
	errorRole	Adapter	Check the configuration of the application that is publishing the event and make sure that it matches the inbound event definition for the above subscription service. For details, see the User's Guide.
AEADCOM_920006	Subscription error. Subscription service <subscription service> listening on subject <subject> received error <error> in SDK message level UserExit <userexit>		
	errorRole	Adapter	Make sure the UserExit parameters are valid and the user exit is invocable from SDK.

Status Code	Role	Category	Resolution
AEADCOM_920007	Subscription error. Subscription service <subscription service> listening on subject <subject> could not get the class description of <class>. The Repository URL is <repoURL> and the Configuration URL is <configURL>.		
	errorRole	Adapter	Please check the repository configuration for this service. For details, see the User’s Guide.
AEADCOM_920008	Subscription error. Subscription service <servicename> listening on subject <subject> could not find the mandatory property <property> in class <class>. The Repository URL is <repoURL> and the Configuration URL is <configURL>		
	errorRole	Adapter	Check the configuration of the application that is publishing the event and make sure that it matches the inbound event definition for the above subscription service. For details, see the User’s Guide.
AEADCOM_920009	Subscription error. Subscription service <servicename> listening on subject <subject> received event with invalid value <value> for property <property> in class <class>.		
	errorRole	Adapter	Check the configuration of the application that is publishing the event and make sure that it matches the inbound event definition for the above subscription service. For details, see the User’s Guide.
AEADCOM_920010	Subscription error. Subscription service <servicename> listening on subject <subject> received event with missing attribute <attribute> in class <class>. The Repository URL is <repoURL> and the Configuration URL is <configURL>.		
	errorRole	Adapter	Check the configuration of the application that is publishing the event and make sure that it matches the inbound event definition for the above subscription service. For details, see the User’s Guide.

Status Code	Role	Category	Resolution
AEADCOM_920011	Subscription error. Subscription service <service name> listening on subject <subject> received event with missing association <association> for class <class>. The Repository URL is <repoURL> and the Configuration URL is <configURL>.		
	errorRole	Adapter	Check the configuration of the application that is publishing the event and make sure that it matches the inbound event definition for the above subscription service. For details, see the User's Guide.
AEADCOM_920012	Subscription error. Subscription service <servicename> listening on subject <subject> received MBusinessDocument <document> with NULL value for attribute <attribute>.		
	errorRole	Adapter	Check the configuration of the application that is publishing the event and make sure that it matches the inbound event definition for the above subscription service. For details, see the User's Guide.
AEADCOM_920013	Subscription error. Subscription service <servicename> listening on subject <subject> received MBusinessDocument <document> with invalid value <value> for attribute <attribute>.		
	errorRole	Adapter	Check the configuration of the application that is publishing the event and make sure that it matches the inbound event definition for the above subscription service. For details, see the User's Guide.

Status Code	Role	Category	Resolution
AEADCOM_920014	Subscription error. Subscription service <service name> listening on subject <subject> could not process the inbound event due to connection error <error> against target application <application> with parameters <parameters> after <number> connection retries. The connection timeout is <timeout_value>.		
	errorRole	Adapter	Check the target application and make sure it is up and running. Check the validity of the connection parameters. For details, see the User's Guide.
AEADCOM_920015	Subscription error. Subscription service <servicename> listening on subject <subject> failed due to target application invocation error <error>. Target application is <application>. The target application specific commands and parameters are <parameters>		
	errorRole	Adapter	Check the target application command and the parameters and make sure they are valid.
AEADCOM_920016	Subscription error. Subscription service <servicename> listening on <port> received error <error> in Post Processing UserExit <userexit>. The user exit parameters are <parameters>		
	errorRole	Adapter	
AEADCOM_920017	Subscription error. Subscription service <service name> listening on <port> could not send response <response> on reply subject <subject>. The parameters for publisher endpoint for sending the reply are <parameters>. Repository URL is <repoURL> and Configuration URL is <configURL>		
	errorRole	Adapter	Please check your repository settings for the publish endpoint of this subscription service. For details, see the User's Guide.

Status Code	Role	Category	Resolution
AEADCOM_920018	Subscription error. Subscription service <service name> listening on <subject>. could not send target application invocation error <error> on error subject <subject>. The parameters for publisher endpoint for sending the reply are <parameters>		
	errorRole	Adapter	Please check your repository settings for the publish endpoint of this subscription service. For details, see the User's Guide.
AEADCOM_920019	Subscription error. Subscription service <service name> listening on <port> could not create reply from target application response <response> for publish on reply subject <subject>. The parameters for publisher endpoint for sending the reply are <parameters>		
	errorRole	Adapter	Please check your repository settings for the publish endpoint of this subscription service. For details, see the User's Guide.
AE_ADCOM_SUB_100	MExceptionEvent received by subscriber <subscriber>: <error>.		
	errorRole	Adapter	Problem in receiving the data. Check for validity of the incoming data from Interceptor or from external source.
AE_ADCOM_SUB_101	The MSubscriber <subscriber> received a message containing an unsupported complex type (<Type>).		
	errorRole	Adapter	This version of the adapter does not support the type used. Use only the supported types.
AE_ADCOM_SUB_102	The MSubscriber <subscriber> received a message containing a non-matching complex type (<Type>).		
	errorRole	Adapter	This version of the adapter does not support the type used. Use only the supported types.

Status Code	Role	Category	Resolution
AE_ADCOM_SUB_103	Cannot create MInstance from event received by MSubscriber <subscriber> (<HRESULT>).		
	errorRole	Adapter	Problem with the Subscriber. Reconfigure the subscriber correctly in the repository.
AEADCOM_930001	Publication error. Publication service <servicename> publishing on subject <subject> encountered error <error> while trying to connect to target application <application>. Connection parameters are <parameters>, the connection timeout is <timeout> milliseconds, and the number of retry efforts is <number>. Polling timeout is <number> milliseconds.		
	errorRole	Adapter	Check the target application and make sure it is up and running. Check the connection parameters for right syntax and values. For details, see the User's Guide..
AEADCOM_930002	Publication error. Publication service <servicename> with publication subject <subject> encountered error <error> while trying to create publish event with schema <schema>. The Target application details are <application>.		
	errorRole	Adapter	Make sure that the publication service is configured properly.
AEADCOM_930003	Publication error. Publication service <servicename> with publishing subject as <subject> received event from target application <application>. It failed while converting event to "Minstance" as it could not get the class description for <class>. Repository URL is <repoURL> and the Configuration URL is <configURL>.		
	errorRole	Adapter	Please verify the configuration of the publication service and check that the schema/class definitions are present in the repository. For details, see the User's Guide.

Status Code	Role	Category	Resolution
AEADCOM_930004	Publication error. Publication service <servicename> with publishing subject <subject> received the event from target application <application>. It failed while converting event to "Minstance" as it could find property <property> in class <class>. Repository URL is <repoURL> and the Configuration URL is <configURL>.		
	errorRole	Adapter	Please verify the configuration of the publication service and check that the schema definitions are present in the repository. For details, see the User's Guide.
AEADCOM_930005	Publication error. Publication service <servicename> with publishing subject <subject> received the event from target application <application>. It failed while converting event to "Minstance" as property <property> of class <class> has invalid value <value>. Repository URL is <repoURL> and the Configuration URL is <configURL>.		
	errorRole	Adapter	Please verify the configuration of the publication service and check that the schema definitions are present in the repository. For details, see the User's Guide.
AEADCOM_930006	Publication error. Publication service <servicename> with publishing subject <subject> received the event from target application <application>. It failed while converting event to "Minstance" attribute <attribute> of class <class> is missing. Repository URL is <repoURL> and the Configuration URL is <configURL>.		
	errorRole	Adapter	Please verify the configuration of the publication service and check that the schema definitions are present in the repository. For details, see the User's Guide..

Status Code	Role	Category	Resolution
AEADCOM_930007	Publication error. Publication service <servicename> with publication subject <subject> received event from target application but could not create the business document <document>. The target application details are <application>, the Repository URL is <repoURL> and the Configuration URL is <configURL>		
	errorRole	Adapter	Please verify the configuration of the publication service and check that the schema definition for the MbusinessDocument maps properly to the event received from the target application. For details, see the User's Guide.
AEADCOM_930008	Publication error. Publication service <servicename> with publication subject <subject> received SDK Exception <exception> while converting the event received from target application to BusinessDocument. The exception occurred while setting the attribute <attribute> with value of <value> for Business Document <document>. The target application details are <application>, the Repository URL is <repoURL> and Configuration URL is <configURL>.		
	errorRole	Adapter	Please verify the configuration of the publication service and check that the schema definition for the MbusinessDocument maps properly to the event received from the target application. For details, see the User's Guide.
AEADCOM_930009	Publication error. Publication service <servicename> with publication subject <subject> received an event from the target application but encountered error <error> in pre-processing user exit invocation. The User exit is <userexit> and the user exit parameters are <parameters>. The target application details are <application>.		
	errorRole	Adapter	Make sure that the parameters passed to UserExit are valid and the User Exit is invocable by the adapter.

Status Code	Role	Category	Resolution
AEADCOM_930010	Publication error. Publication service <servicename> with publication subject <subject> received error <error> in RvRPC user exit 2-way invocation. The User exit is <userexit>, the user exit endpoint details are <userexit> and the user exit parameters are <parameters>. The user exit timeout is <timeout>.		
	errorRole	Adapter	Please check the validity of User exit endpoint and the parameters passed to the user exit.
AEADCOM_930011	Publication error. Publication service <servicename> with publication subject <subject>, received timeout error in RvRPC user exit one-way invocation. The User exit is <userexit>, the user exit endpoint details are <endpoint> and the user exit parameters are <parameters>. The user exit timeout is <number> milliseconds.		
	errorRole	Adapter	Please check the validity of User exit endpoint and the parameters passed to the user exit.
AEADCOM_930012	Publication error. Publication service ,<servicename> with publication subject <subject> received error <error> in the SDK message level UserExit. The User exit names are <userexit> and the User exit parameters are <parameters>.		
	errorRole	Adapter	Make sure the UserExit parameters are valid and the user exit is invokable from SDK.
AEADCOM_930013	Publication error. Publication service <servicename> with publication subject <subject> received error <error> in post-processing user exit invocation. The User exit is <userexit> and the user exit parameters are <parameters>.		
	errorRole	Adapter	Make sure that the parameters passed to UserExit are valid and the User Exit is invokable by the adapter.

Status Code	Role	Category	Resolution
AEADCOM_930014	Publication error. Publication service <servicename> with publication subject <subject> received error while sending event over the wire. The Publish endpoint details are <details>		
	errorRole	Adapter	Please check repository settings for valid configuration of the publish endpoint for this service.For details, see the User’s Guide.
AE_ADCOM_PUB_15	Publication error: Error occurred while trying to publish. The exception is: <exception>.		
	errorRole	Adapter	Restart the process.
AEADCOM_940001	Request Response error. Request Response service <service name> listening on <subject> received unexpected null data in incoming request. Expects event <events>. The Repository URL is <repoURL> and the Configuration URL is <configURL>		
	errorRole	Adapter	Please check the configuration of the application that is requesting the event and make sure that it matches the inbound event definition for the above RequestResponse service. For details, see the User’s Guide.
AEADCOM_940002	Request Response error. RequestResponse service <servicename> with RequestResponse subject <subject> received an event from the target application but encountered error <error> in pre-processing user exit invocation. The User exit is <userexit> and the user exit parameters are <parameters>. The target application details are <application>.		
	errorRole	Adapter	Please make sure that the parameters passed to User Exit are valid and the User Exit is invocable by the adapter.

Status Code	Role	Category	Resolution
AEADCOM_940003	Request Response error. Request Response service <servicename> with publication subject <subject> received error <error> in the SDK message level User Exit. The User exit names are <userexit> and the User exit parameters are <parameters>.		
	errorRole	Adapter	Please make sure the User Exit parameters are valid and the User Exit is invocable from SDK.
AEADCOM_940005	Request Response error. Request Response service <service name> failed to deserialize the received MServerRequest to MInstance: Received event on subject <subject>, event = <event>, SDK exception = <exception>. The Repository URL is <repoURL> and the Configuration URL is <configURL>		
	errorRole	Adapter	Please check the configuration of the application that is requesting the event and make sure that it matches the inbound event definition for the above Request Response service. For details, see the User's Guide.
AEADCOM_940006	Request Response error. Error in incoming data for RPC service: <servicename> on subject: <subject>. Not able to map incoming data to the business object schema: <schema> configured at design time. Invalid data type: <Type> for attribute: <attribute>		
	errorRole	Adapter	Please check the configuration of the application that is requesting the event and make sure that it matches the inbound event definition for the above Request Response service. For details, see the User's Guide.
AEADCOM_940007	Request Response error. Error in incoming data for RPC service: <servicename> on subject: <subject>. Missing mandatory parameter <parameter> for RPC input class <class>		
	errorRole	Adapter	Please check the configuration of the application that is requesting the event and make sure that it matches the inbound event definition for the above Request Response service. For details, see the User's Guide.

Status Code	Role	Category	Resolution
AEADCOM_940008	Request Response error. Connection error in invocation of RPC service:<servicename> on subject:<subject>. Connection parameters are parameter_name: <name> parameter_value:<value>		
	errorRole	Adapter	Check if the end application is up and running. Also verify the connection parameters that are specified in the repository.
AEADCOM_940009	Request Response error. Request Response service <servicename> listening on subject <subject> failed due to target application invocation error <error>. Target application is <application> and inbound event is <event>. <Where applicable add Target application specific commands and parameters are <parameters>		
	errorRole	Adapter	Please check the target application command and the parameters and make sure they are valid.
AEADCOM_940010	Request Response error. Request Response service <servicename> listening on subject <subject> failed to create Reply Business Object Error <error>.		
	errorRole	Adapter	Please check the target application command and the parameters and make sure they are valid.
AEADCOM_940011	Request Response error. Request Response service <servicename> listening on subject <subject> receive a time out error. Time out period in configuration file is <timeout>.		
	errorRole	Adapter	Please check the target application command and the parameters and make sure they are valid.
AEADCOM_940012	Request Response error. Request Response service <service name> listening on subject <subject> receive an error while sending Data on Reply Address <address>. Error Message <message>		
	errorRole	Adapter	Please check whether the request client is alive.

Status Code	Role	Category	Resolution
AEADCOM_940013	Request Response error. Request Response service <servicename> with subject <subject> received error <error> in post-processing user exit invocation. The User exit is <userexit> and the user exit parameters are <parameters>.		
	errorRole	Adapter	Please make sure that the parameters passed to User Exit are valid and the User Exit is invocable by the adapter.
AEADCOM_950001	Request Response Invocation error. Request Response Invocation service <servicename> with subject as <subject> received event from target application <application>. It failed while converting event to Request, as it could not get the class description for <class>. Repository URL is <repoURL> and the Configuration URL is <configURL>.		
	errorRole	Adapter	Please verify the configuration of the Request Response Invocation Service and check that the schema/class definitions are present in the repository. For details, see the User's Guide.
AEADCOM_950002	Request Response Invocation error. Request Response Invocation service <servicename> with subject as <subject> received event from target application <application>. It failed while converting event to Request, as it could not get the Operation description <description> in the class <class>. Repository URL is <repoURL> and the Configuration URL is <configURL>.		
	errorRole	Adapter	Please verify the configuration of the Request Response Invocation Service and check that the schema/class definitions are present in the repository. For details, see the User's Guide.

Status Code	Role	Category	Resolution
AEADCOM_950003	Request Response Invocation error. Request Response Invocation service <servicename> with subject as <subject> received event from target application <application>. It failed while converting event to Request. Repository URL is <repoURL> and the Configuration URL is <configURL>.		
	errorRole	Adapter	Please verify the configuration of the Request Response Invocation Service and check that the schema/class definitions are present in the repository. For details, see the User’s Guide.
AEADCOM_950004	Request Response Invocation error. Request Response Invocation service <servicename> with subject as <subject> received event from target application <application>. It failed while converting event to Request, as it cannot set Operation Parameter % 4 <parameter>. Repository URL is <repoURL> and the Configuration URL is <configURL>.		
	errorRole	Adapter	Please verify the configuration of the Request Response Invocation Service and check that the schema/class definitions are present in the repository. For details, see the User’s Guide.
AEADCOM_950005	Request Response Invocation error. Request Response Invocation service <servicename> with Request Response Invocation subject <subject> received an event from the target application but encountered error <error> in pre-processing user exit invocation. The User exit is <userexit> and the user exit parameters are <parameters>. The target application details are <application>.		
	errorRole	Adapter	Please make sure that the parameters passed to User Exit are valid and the User Exit is invocable by the adapter.

Status Code	Role	Category	Resolution
AEADCOM_950006	Request Response Invocation error. Request Response Invocation <servicename> with subject <subject> received error <error> in the SDK message level User Exit. The User exit names are <userexit> and the User exit parameters are <parameters>.		
	errorRole	Adapter	Make sure the User Exit parameters are valid and the user exit is invocable from SDK.
AEADCOM_950007	Request Response Invocation error. Request Response Invocation service <servicename> with subject <subject> received error while sending event over the wire. The Request Response Invocation endpoint details are <endpoint>		
	errorRole	Adapter	Please check repository settings for valid configuration of the Request Response Invocation endpoint for this service. For details, see the User's Guide.
AEADCOM_950008	Request Response Invocation error. Request Response Invocation service <servicename> with subject <subject> received error while requesting event over the wire. The Request Response invocation endpoint details are <endpoint>		
	errorRole	Adapter	Please check repository settings for valid configuration of the Request Response Invocation endpoint for this service. For details, see the User's Guide.
AEADCOM_950009	Request Response Invocation error. Request Response Invocation service <servicename> with subject <subject> received error while requesting event over the wire. The Request Response Invocation endpoint details are <endpoint>		
	errorRole	Adapter	Please check repository settings for valid configuration of the Request Response Invocation endpoint for this service. For details, see the User's Guide.

Status Code	Role	Category	Resolution
AEADCOM_950010	Request Response Invocation error. Request Response Invocation service <servicename> with subject <subject> received null reply while requesting event over the wire. The Request Response Invocation endpoint details are <endpoint>		
	errorRole	Adapter	Please check the target application, if it is running or not. Please check the configuration of request response Invocation service.
AEADCOM_950011	Request Response Invocation error. Request Response Invocation service <servicename> with subject <subject> received timeout error while requesting event over the wire. The Request Response Invocation endpoint details are <endpoint>		
	errorRole	Adapter	Please check repository settings for valid configuration of the Request Response Invocation endpoint for this service. For details, see the User's Guide.
AEADCOM_950012	Request Response Invocation error. Request Response Invocation service <servicename> with subject <subject> received error while processing reply message. The Request Response Invocation endpoint details are <endpoint>		
	errorRole	Adapter	Please check repository settings for valid configuration of the Request Response Invocation endpoint for this service. For details, see the User's Guide.
AEADCOM_950013	Request Response Invocation error. Request Response Invocation service <servicename> with subject <subject>, received error <error> in post-processing user exit invocation. The User exit is <userexit> and the user exit parameters are <parameters>.		
	errorRole	Adapter	Please make sure that the parameters passed to User Exit are valid and the User Exit is invocable by the adapter.

Status Code	Role	Category	Resolution
AE_ADCOM_SYSTEM_100	System error. Unknown exception encountered.		
	errorRole	Adapter	Restart the adapter.
AE_ADCOM_SYSTEM_101	System error. Invalid interface pointer.		
	errorRole	Adapter	Make sure that the registered DLL is in its physical location.
AE_ADCOM_SYSTEM_102	System error. Bad service request.		
	errorRole	Adapter	Problem with the Interceptor service. Please reinstall it.
AE_ADCOM_SYSTEM_103	System error. Advisory: <advisory> : <message>.		
	errorRole	Adapter	Check the actual error message for details.
AE_ADCOM_SYSTEM_104	System error. Advisory: <advisory> : <message>.		
	errorRole	Adapter	Check the actual error message for details.
AE_ADCOM_SYSTEM_105	MException thrown: <exception>.		
	errorRole	Adapter	Check error description in the event viewer for more information.
AE_ADCOM_SYSTEM_106	Unknown exception thrown by implementation of coclass <class> (interface <Interface>, method <method>).		
	errorRole	Adapter	Make sure that the registered DLL is in its physical location.
AE_ADCOM_SYSTEM_107	Cannot create event dispatching thread.		
	errorRole	Adapter	Problem with creating the thread. Restart the process and system (if possible).
AE_ADCOM_SYSTEM_108	<description>		
	errorRole	Adapter	Check for the previous error messages.

Status Code	Role	Category	Resolution
AE_ADCOM_SYSTEM_110	RPC server Unavailable. Adapter lost COM server Connection.		
	errorRole	System	RPC server unavailable. Check whether the COM server process is running.
AE_ADCOM_SYSTEM_111	COM server Reconnection attempts Failed. Suspending Subscriber Service: <servicename> with subject <subject>.		
	infoRole	System	RPC server unavailable. Check whether the COM server process is running.
AE_ADCOM_SYSTEM_112	Successfully Reconnected to COM server. Activating Subscriber Service: <servicename> with subject <subject>.		
	infoRole	System	Indicates normal adapter information. No action necessary
AE_ADCOM_SYSTEM_113	COM server Reconnection attempts Failed. Suspending RPC Service: <servicename> with subject <subject>.		
	infoRole	System	RPC server unavailable. Check whether the COM server process is running.
AE_ADCOM_SYSTEM_114	Successfully Reconnected to COM server. Activating RPC Service: <servicename> with subject <subject>.		
	infoRole	System	Indicates normal adapter information. No action necessary.
AEADCOM_990001	Shutdown error. Failed to deactivate the <name> timer. SDK exception = <exception>		
	errorRole	Adapter	Check for the SDK exception details.
AEADCOM_990002	Shutdown error. SDK cleanup exception = <exception>		
	errorRole	Adapter	Check for the SDK exception details.
AEADCOM_990003	Shutdown error. Failed to cleanup Hawk microagent. SDK exception = <exception>		
	errorRole	Adapter	Check for the SDK exception details.

Status Code	Role	Category	Resolution
AEADCOM_990004	Shutdown error. Failed to cleanup the threads: <error>		
	errorRole	Adapter	Restart adcomService.
AEADCOM_990005	Shutdown error. Error in disconnecting from <name> connection parameters <name>.		
	errorRole	Adapter	Restart adcomService.
AE_ADCOM_SHUT_100	Shutdown error. Wait for TIBCOCOMService stop event failed.		
	errorRole	Adapter	Restart adcomService.
AE_ADCOM_SHUT_101	Shutdown error. Call to SetEvent failed.		
	errorRole	Adapter	Restart adcomService.
AE_ADCOM_SHUT_102	Shutdown error. Can't deleteNTservice.		
	errorRole	Adapter	Deleting the service failed. Try deleting the service again.
AE_ADCOM_SHUT_103	Wait for TIBCOCOMService stop event failed.		
	errorRole	Adapter	Restart adcomService.
AE_ADCOM_SHUT_104	Adapter Termination Criteria is set to any service suspended: Terminating adapter service.		
	infoRole	Shutdown	RPC server Unavailable. Check whether the COM server process is running and restart the adapter.
AE_ADCOM_SHUT_105	All Adapter Services suspended: Terminating adapter service.		
	infoRole	Shutdown	RPC server Unavailable. Check whether the COM server process is running and restart the adapter.
AE_ADCOM_SHUT_106	Adapter reconnection attempt (<attempt number>) failed.		
	infoRole	Shutdown	RPC server Unavailable. Check whether the COM server process is running.

Status Code	Role	Category	Resolution
AE_ADCOM_SHUT_107	Adapter failed to reconnect. Terminating adapter service.		
	infoRole	Shutdown	RPC server Unavailable. Check whether the COM server process is running and restart the adapter.

TIBCO Software Inc. End User License Agreement

READ THIS END USER LICENSE AGREEMENT CAREFULLY. BY DOWNLOADING OR INSTALLING THE SOFTWARE, YOU AGREE TO BE BOUND BY THIS AGREEMENT. IF YOU DO NOT AGREE TO THESE TERMS, DO NOT DOWNLOAD OR INSTALL THE SOFTWARE AND RETURN IT TO THE VENDOR FROM WHICH IT WAS PURCHASED. RETURNS BY THE ORIGINAL PURCHASER WITHIN THIRTY (30) DAYS OF THE PURCHASE DATE WILL RECEIVE A FULL REFUND.

Upon your acceptance as indicated above, the following shall govern your use of the Software except to the extent all or any portion of the Software (a) is subject to a separate written agreement, (b) includes a separate "click-on" license agreement as part of the download or installation process, or (c) is provided by a third party under the terms set forth in an Addenda at the end of this Agreement, in which case the terms of such addenda shall control over inconsistent terms with regard to such portion(s).

License Grant. The Software is the property of TIBCO or its licensors and is protected by copyright and other laws. While TIBCO continues to own the Software, TIBCO hereby grants to Customer a limited, non-transferable, non-exclusive, license to use the number of Permitted Instances set forth in the Ordering Document, in machine-readable, object code form and solely for Customer's internal business use.

Restrictions. Customer agrees not to (a) make more copies than the number of Permitted Instances plus a reasonable number of backups; (b) provide access to the Software to anyone other than employees, contractors, or consultants of Customer; (c) sublicense, transfer, assign, distribute to any third party, pledge, lease, rent, or commercially share the Software or any of Customer's rights under this Agreement (for the purposes of the foregoing a change in control of Licensee is deemed to be an assignment); (d) use the Software for purposes of providing a service bureau, including, without limitation, providing third-party hosting, or third-party application integration or application service provider-type services, or any similar services; (e) use the Software in connection with ultrahazardous activities, or any activity for which failure of the Software might result in death or serious bodily injury to Customer or a third party; or (f) directly or indirectly, in whole or in part, modify, translate, reverse engineer, decrypt, decompile, disassemble, make error corrections to, create derivative works based on, or otherwise attempt to discover the source code or underlying ideas or algorithms of the Software.

Beta and Evaluation Licenses. Notwithstanding the foregoing, if the Software is being provided for demonstration, beta testing, or evaluation purposes, then Customer agrees (a) to use the Software solely for such purposes, (b) that the Software will not be used or deployed in a production environment, and (c) that such use shall automatically terminate upon the earlier of thirty days from the date Customer receives the right to install the Software, or Customer's receipt of notice of termination from TIBCO.

Technical Support. Provided Customer has paid applicable support fees (not included with Software fees unless separately listed), TIBCO shall provide support for generally available TIBCO Software on an annual basis commencing on the Purchase Date, as follows ("Support"): Customer shall designate at TIBCO's support website <https://support.tibco.com/eSupport/newuser.html>, the number of technical support contacts permitted under the level of Support purchased (contacts are changeable upon 48-hours prior written notice to TIBCO). Each contact may contact TIBCO for problem

resolution during TIBCO's published support hours corresponding to the level of Support fees paid.

Upon notice from a contact of a Software problem which can be reproduced at a TIBCO support facility or via remote access to Customer's facility, TIBCO shall use reasonable efforts to correct or circumvent the problem according to its published support objectives. TIBCO reserves the right to make changes only to the most currently available version. TIBCO will use reasonable efforts to support the previously released version of the Software for a maximum of six months.

TIBCO shall have no obligation to support the Software (i) for use on any computer system running other than the operating system software for which the Software is approved (as set forth in the Software documentation) and licensed hereunder, or (ii) if Customer has modified or authorized a third party to modify the Software. TIBCO shall have no obligation to modify any version of the Software to run with any new versions of any operating system, or any other third party software or hardware. If Customer purchases Support for any Software, Customer must purchase the same level of Support for all copies of the Software for which it is licensed.

Support may be extended for one-year periods on the anniversary of each Purchase Date at the standard amounts set forth in its price list, for as long as TIBCO offers Support. Customer may reinstate lapsed support for any then currently supported Software by paying all Support fees in arrears and any applicable reinstatement fee. Upgrades, patches, enhancements, bug fixes, new versions and/or new releases of the Software provided from time to time under Support shall be used only as replacements to existing Permitted Instances, and shall not be deemed to increase that number, and use thereof shall be governed by the terms of this Agreement, except for the first paragraph of the Limited Warranty and any right of return or refund.

Consulting Services. Customer may request additional services ("Services") either in an Ordering Document, or by a separate mutually executed work order, statement of work or other work-request document incorporating this Agreement (each, a "Work Order"). Unless otherwise expressly agreed to in a Work Order, all Services and any work product therefrom shall be (a) performed on a time and materials basis, plus meals, lodging, travel, and other expenses reasonably incurred in connection therewith, (b) deemed accepted upon delivery, and (c) exclusively owned by TIBCO (except for confidential information of Customer identified to TIBCO in the Ordering Document), including all right, title and intellectual property or other right or interest therein. Each Work Order is intended to constitute an independent and distinct agreement of the parties, notwithstanding that each shall be construed to incorporate all applicable provisions of this Agreement. Specific to TIBCO training services, additional information regarding courses, registration, restrictions or limitation can be found at TIBCO's website at <http://www.tibco.com/services/education> under Education Programs. Fees for Services shall be due and payable in United States dollars net 30 from the date of TIBCO's invoice.

Limited Warranty. If Customer obtained the Software directly from TIBCO, then TIBCO warrants that for a period of thirty (30) days from the Purchase Date: (i) the media on which the Software is furnished will be free of defects in materials and workmanship under normal use; and (ii) the Software will substantially conform to its published specifications. This limited warranty extends only to the original Customer hereunder. Customer's sole and exclusive remedy and the

entire liability of TIBCO and its suppliers under this limited warranty will be, at TIBCO's option, repair, replacement, or refund of the Software and applicable Support fees, in which event this Agreement shall terminate upon payment thereof.

This warranty does not apply to any Software which (a) is licensed for beta, evaluation, testing or demonstration purposes for which TIBCO does not receive a license fee, (b) has been altered or modified, except by TIBCO, (c) has not been installed, operated, repaired, or maintained in accordance with instructions supplied by TIBCO, (d) has been subjected to abnormal physical or electrical stress, misuse, negligence, or accident, or (e) is used in violation of any other term of this Agreement. Customer agrees to pay TIBCO for any Support or Services provided by TIBCO related to a breach of the foregoing on a time, materials, travel, lodging and other reasonable expenses basis. If Customer obtained the Software from a TIBCO reseller or distributor, the terms of any warranty shall be as provided by such reseller or distributor, and TIBCO provides Customer no warranty with respect to such Software.

EXCEPT AS SPECIFIED IN THIS LIMITED WARRANTY, THE SOFTWARE, SUPPORT AND SERVICES ARE PROVIDED "AS IS", ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS, AND WARRANTIES INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTY OR CONDITION OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NONINFRINGEMENT, SATISFACTORY QUALITY OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE, ARE HEREBY EXCLUDED TO THE EXTENT ALLOWED BY APPLICABLE LAW. NO WARRANTY IS MADE REGARDING THE RESULTS OF ANY SOFTWARE, SUPPORT OR SERVICES OR THAT THE SOFTWARE WILL OPERATE WITHOUT ERRORS, PROBLEMS OR INTERRUPTIONS, OR THAT ERRORS OR BUGS IN THE SOFTWARE WILL BE CORRECTED, OR THAT THE SOFTWARE'S FUNCTIONALITY OR SERVICES WILL MEET CUSTOMER'S REQUIREMENTS. NO TIBCO DEALER, DISTRIBUTOR, AGENT OR EMPLOYEE IS AUTHORIZED TO MAKE ANY MODIFICATIONS, EXTENSIONS OR ADDITIONS TO THIS WARRANTY.

Indemnity. If Customer obtained the Software from TIBCO directly, then TIBCO shall indemnify Licensee from and against any final judgment by a court of competent jurisdiction, including reasonable attorneys' fees, that the unmodified TIBCO Software infringes any patent issued by the United States, Canada, Australia, Japan, or any member of the European Union, or any copyright, or any trade secret of a third party; provided that TIBCO is promptly notified in writing of such claim, TIBCO has the exclusive right to control such defense and/or settlement, and Licensee shall provide reasonable assistance (at TIBCO's expense) in the defense thereof. In no event shall Licensee settle any claim, action or proceeding without TIBCO's prior written approval. In the event of any such claim, litigation or threat thereof, TIBCO, at its sole option and expense, shall (a) procure for Licensee the right to continue to use the TIBCO Software or (b) replace or modify the TIBCO Software with functionally equivalent software. If such settlement or modification is not commercially reasonable (in the reasonable opinion of TIBCO), TIBCO may cancel this Agreement upon sixty days prior written notice to Licensee, and refund to Licensee the unamortized portion of the license fees paid to TIBCO by Licensee based on a five-year straight-line depreciation. This Section states the entire liability of TIBCO with respect to the infringement of any Intellectual Property rights, and Licensee hereby expressly waives any other liabilities or obligations of TIBCO with respect thereto. The foregoing indemnity shall not apply to the extent any infringement could have been avoided by use of the then-current release.

Limitation of Liability. EXCEPT AS PROVIDED UNDER INDEMNITY OR RESULTING FROM A BREACH OF CONFIDENTIALITY (THE "EXCLUDED MATTERS"), IN NO EVENT WILL EITHER PARTY BE LIABLE FOR ANY LOST DATA, LOST REVENUE, LOST PROFITS, DAMAGE TO REPUTATION, BUSINESS INTERRUPTION, OR ANY OTHER INDIRECT, INCIDENTAL, CONSEQUENTIAL, SPECIAL, PUNITIVE, EXEMPLARY OR ANY SIMILAR TYPE DAMAGES ARISING OUT OF THIS AGREEMENT, THE USE OR THE INABILITY TO USE THE SOFTWARE, OR THE PROVISION OF ANY SUPPORT OR SERVICES, EVEN IF A PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. EXCEPT FOR THE EXCLUDED MATTERS, IN NO EVENT SHALL A PARTY BE LIABLE TO THE OTHER, WHETHER IN CONTRACT, TORT (INCLUDING ACTIVE OR PASSIVE NEGLIGENCE), BREACH OF WARRANTY, CLAIMS BY THIRD PARTIES OR OTHERWISE, EXCEED THE PRICE PAID BY CUSTOMER UNDER THE APPLICABLE ORDERING DOCUMENT.

THE FOREGOING LIMITATIONS SHALL APPLY EVEN IF THE ABOVE-STATED REMEDY OR LIMITED WARRANTY FAILS OF ITS ESSENTIAL PURPOSE. BECAUSE SOME STATES OR JURISDICTIONS DO NOT ALLOW LIMITATION OR EXCLUSION OF CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATION MAY NOT APPLY TO CUSTOMER.

Confidentiality. "Confidential Information" means the terms of this Agreement; all information marked by the disclosing party as proprietary or confidential; any provided software, related documentation or related performance test results derived by Licensee; and any methods, concepts or processes utilized in provided software or related documentation. Confidential Information shall remain the sole property of the disclosing party and shall not be disclosed to any non-Authorized User without the prior written consent of the disclosing party. If Confidential Information is communicated orally, such communication shall be confirmed as "Confidential" in writing within thirty days of such disclosure. The parties agree to protect the Confidential Information of the other in the same manner it protects the confidentiality of similar information and data of its own (and at all times exercising at least a reasonable degree of care). Except with respect to the Software, items will not be deemed Confidential Information if (i) available to the public other than by a breach of an agreement with TIBCO, (ii) rightfully received from a third party not in breach of any obligation of confidentiality, (iii) independently developed by one party without use of the Confidential Information of the other; (iv) known to the recipient at the time of disclosure (other than under a separate confidentiality obligation); or (v) produced in compliance with applicable law or court order, provided the other party is given reasonable notice of the same. Both parties agree to indemnify the other for any damages the other may sustain resulting from their unauthorized use and/or disclosure of the other's Confidential Information. Such damages shall include reasonable expenses incurred in seeking both legal and equitable remedies. To the extent required by law, at Customer's request, TIBCO shall provide Customer with the interface information needed to achieve interoperability between the Software and another independently created program, on payment of TIBCO's applicable fee. Customer agrees to observe obligations of confidentiality with respect to such information.

Export. Software, including technical data, is subject to U.S. export control laws, including the U.S. Export Administration Act and its associated regulations, and may be subject to export or import regulations in other countries. Customer agrees to comply strictly with all such regulations and agrees to obtain all necessary licenses to export, re-export, or import Software.

Government Use. If the Customer is an agency, department, or other entity of the United States Government ("Government"), the use, duplication, reproduction, release, modification, disclosure or transfer of the Software, or any related documentation of any kind, including technical data or manuals, is restricted in accordance with Federal Acquisition Regulation ("FAR") 12.212 for civilian agencies and Defense Federal Acquisition Regulation Supplement ("DFARS") 227.7202 for military agencies. The Software is commercial computer software and commercial computer software documentation. Use of the Software and related documentation by the Government is further restricted in accordance with the terms of this Agreement, and any modification thereto.

Orders. An Ordering Document shall be deemed accepted only by issuance of a TIBCO invoice and solely for purposes of administrative convenience. None of the terms of the Ordering Document (other than the Software product name, number of Permitted Instances, level of Support, description of Services, and fees due in connection therewith) shall apply for any reason or purpose whatsoever, regardless of any statement on any Ordering Document to the contrary, unless countersigned by an officer of TIBCO. This Agreement constitutes the entire agreement between the parties with respect to the use of the Software, Support and Services, and supersedes all proposals, oral or written, and all other representations, statements, negotiations and undertakings relating to the subject matter hereof. All orders of Software, Support or Services by Customer from TIBCO shall be deemed to occur under the terms of this Agreement (with or without reference to this Agreement), unless expressly superseded by a signed written Agreement between the parties. Software shall be delivered electronically (unless physical shipment is specifically set forth in an Ordering Document), and such delivery shall occur upon e-mail of download information to Licensee at the email address set forth in the Ordering Document or as otherwise provided by Customer to TIBCO. Physical deliveries (if applicable) of Software and documentation which typically accompanies the Software on delivery shall be on CD-ROM, FOB TIBCO, and delivery shall occur by depositing the CD-ROM with TIBCO's overnight carrier (at no charge to Customer).

Term and Termination. Support or Services may be terminated: (a) by either party upon a default of the other, such default remaining uncured for fifteen days from written notice from the non-defaulting party; (b) upon the filing for bankruptcy or insolvency of the other party, (c) by either party upon prior written notice at least sixty days prior to the end of any annual Maintenance period; or (d) by Licensee (for Services), upon ten days prior written notice. Termination of Support or Services shall not terminate this Agreement. Customer may terminate this Agreement in its entirety at any time by destroying all copies of the Software. Upon termination of this Agreement in its entirety, for any reason, Customer must cease using and return or destroy all copies of the Software. Customer's obligation to pay accrued charges and any fees due as of the date of termination, as well as the sections entitled "Confidentiality", "Limited Warranty" and "Limitation of Liability" shall survive any such termination.

Authority. You hereby represent and warrant that you have full power and authority to accept the terms of this Agreement on behalf of Customer, and that Customer agrees to be bound by this Agreement.

General. Fees on the Ordering Document (all to be paid on the latter of thirty days from Invoice by TIBCO or the date set forth in the Ordering Document) do not include sales, use, withholding, value-added or similar taxes, and Customer agrees to pay the same, excluding therefrom taxes related to TIBCO's income and corporate franchise tax. Customer agrees to pay all reasonable costs incurred (including reasonable attorneys' fees) in collecting past due amounts under this Agreement. Except as set forth in the Section entitled

"Limited Warranty" all fees paid under or in connection with this Agreement are non-refundable and no right of set-off exists. All payments of fees due shall be made in U.S. dollars, net 30 from Purchase Date, or, for any other amounts coming due hereafter, net 30 from TIBCO's invoice. A service charge of one and one-half percent per month will be applied to all invoices that are not paid on time. Licensee agrees to pay all sales, use, value-added, withholding, excise and any other similar taxes or government charges, exclusive of TIBCO's income taxes. No delay in the performance of any obligation by either party, excepting all obligations to make payment, shall constitute a breach of this Agreement to the extent caused by force majeure. Customer hereby grants TIBCO and its independent auditors the right to audit Customer's compliance with this Agreement. If any portion of this Agreement is found to be void or unenforceable, the remaining provisions shall remain in full force and effect. This Agreement shall be governed by and construed in accordance with the laws of the State of California, United States of America, as if performed wholly within the state and without giving effect to the principles of conflict of law. The state and/or federal courts in San Francisco, California, USA, shall have exclusive jurisdiction of any action arising out of or relating to this Agreement. The United Nations Convention on Contracts for the International Sale of Goods is excluded from application hereto. If any portion hereof is found to be void or unenforceable, the remaining provisions of this Agreement shall remain in full force and effect.

Definitions. In connection with this Agreement, the following capitalized terms shall have the following meaning: "Agreement" means this End User License Agreement; "Connection" for the TIBCO Software product TIBCO Enterprise for JMS - Full Edition means a TIBCO Enterprise for JMS client connection to the TIBCO Enterprise for JMS server for the purpose of sending or receiving messages and for the purposes of the TIBCO Software products TIBCO SmartSockets and TIBCO SmartMQ software products, a Connection means any network protocol link established with such TIBCO Software (directly or indirectly) to any other entity, including but not limited to software, firmware or hardware; "Customer" means the original purchaser or licensee of the Software and any permitted successors and assigns; "Developer" means one user/developer of a TIBCO Software product for use in Development; "Development" means used for software development purposes only; "Enterprise" means an unlimited number of Permitted Instances for a period of one year from the Purchase Date (unless otherwise set forth in the Ordering Document), at which time existing licenses convert to perpetual and Customer may not thereafter deploy additional Permitted Instances, and in any event, shall (during the one-year unlimited deployment period) exclude any entity which acquires, is acquired by, merged into, or otherwise combined with Customer. Customer hereby agrees to provide TIBCO with notice of the number of Permitted Instances deployed at the end of such one-year period within thirty days thereafter; "Fab" means unlimited use for shop-floor manufacturing applications at a Site; "Workstation" shall mean a single end-user computer that is generally intended to be accessed by one person at a time; "Ordering Document" means any purchase order or similar document or agreement requesting Software, Support or Services; "Permitted Instance(s)" means the number of copies of Software running on a Server Instance, Workstation, User, or Development basis, on a designated Platform, as set forth in an Ordering Document, including, without limitation, Enterprise, Site and Fab licensing; "Platform" means the operating system set forth in an Ordering Document; "Purchase Date" means the date the Ordering Document is accepted by TIBCO; "Server Instance" means a computer with 1 CPU (unless otherwise set forth in the Ordering Document) performing common services for multiple machines; "Site" means an unlimited number of Permitted Instances at a specific physical address set forth in the Ordering Document (or, in the absence of any address, at Customer's corporate headquarters);

"Software" means the software products listed in an Ordering Document (except as provided in the second paragraph hereof), in whole and in part, along with their associated documentation; "TIBCO" means TIBCO Software Inc.; and "Named User" means the number of named users with access to the Software.

Special Product Provisions. TIBCO BusinessPartner: Customer may sublicense to third parties ("Partners") up to the total Number of Copies of TIBCO BusinessPartner, provided that for every such sublicense, the Number of Copies Customer is licensed to use shall be reduced by the same number, and provided further that prior to delivery of TIBCO BusinessPartner to a Partner, such Partner agrees in writing (a) to be bound by terms and conditions at least as protective of TIBCO as the terms of this Agreement, (b) that TIBCO BusinessPartner be used solely to communicate with Customer's implementation of TIBCO BusinessConnect, and (c) for such Partner to direct all technical support and Maintenance questions directly to Customer. Customer agrees to keep records of the Partners to which it distributes TIBCO BusinessPartner, and to provide TIBCO the names thereof (with an address and contact name) within sixty days of the end of each quarter. Third Party Software: Use of any other third-party software identified by its company and/or product name or otherwise designated in Licensee's Ordering Document (collectively "Third Party Software") is subject solely to the terms and conditions of the click-wrap or shrink-wrap license agreement included with the Third Party Software products, and for which TIBCO shall be an intended third-party beneficiary of same. TIBCO shall have no obligation whatsoever in connection with the Third Party Software (including, without limitation, any obligation to provide maintenance or support) and the provision of Third Party Software is accomplished solely as an accommodation and in lieu of Customer purchasing a license to Third Party Software directly from the third party vendor. Embedded/Bundled Products: Some TIBCO Software embeds or bundles other TIBCO Software (e.g., TIBCO InConcert bundles TIBCO Rendezvous). Use of such embedded or bundled TIBCO Software is solely to enable the functionality of the TIBCO Software licensed on the Cover Page, and may not be used or accessed by any other TIBCO Software, or for any other purpose. Open Source Software: If Licensee uses Open Source software in conjunction with the TIBCO Software, Licensee must ensure that its use does not (i) create, or purport to create, obligations of use with respect to the TIBCO Software, or (ii) grant, or purport to grant, to any third party any rights to or immunities under TIBCO's intellectual property or proprietary rights in the TIBCO Software. You also may not combine the TIBCO Software with programs licensed under the GNU General Public License ("GPL") in any manner that could cause, or could be interpreted or asserted to cause, the TIBCO Software or any modifications thereto to become subject to the terms of the GPL.

Copyright (c) 1994-2005 TIBCO Software Inc. ALL RIGHTS RESERVED.

ADDENDA: Third Party License Agreements

Third Party Software License Agreements

The following are the software licenses for the Third Party Software provided in connection with the software.

The Apache Software License, Version 1.1

Copyright (c) 1999 The Apache Software Foundation. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The end-user documentation included with the redistribution, if any, must include the following acknowledgment: "This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>).". Alternately, this acknowledgment may appear in the software itself, if and wherever such third-party acknowledgments normally appear.
4. The names "Xerces", "Xalan" and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact apache@apache.org. Products derived from this software may not be called "Apache", nor may "Apache" appear in their name, without prior written permission of the Apache Software Foundation.

* THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

* This software consists of voluntary contributions made by many individuals on behalf of the Apache Software Foundation and was originally based on software copyright (c) 1999, Lotus Development Corporation., <http://www.lotus.com>. For more information on the Apache Software Foundation, please see <http://www.apache.org/>.

The Apache Software License, Version 1.1

Copyright (c) 1999 The Apache Software Foundation. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The end-user documentation included with the redistribution, if any, must include the following acknowledgment:

"This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>).". Alternately, this acknowledgment may appear in the software itself, if and wherever such third-party acknowledgments normally appear.

4. The names "Xalan" and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact apache@apache.org.

5. Products derived from this software may not be called "Apache", nor may "Apache" appear in their name, without prior written permission of the Apache Software Foundation.

THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This software consists of voluntary contributions made by many individuals on behalf of the Apache Software Foundation and was originally based on software copyright (c) 1999, Lotus Development Corporation., <http://www.lotus.com>. For more information on the Apache Software Foundation, please see <http://www.apache.org/>.

W3C IPR SOFTWARE NOTICE

Copyright 2000 World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved.

The DOM bindings are published under the W3C Software Copyright Notice and License. The software license requires "Notice of any changes or modifications to the W3C files, including the date changes were made."

Consequently, modified versions of the DOM bindings must document that they do not conform to the W3C standard; in the case of the IDL

binding, the pragma prefix can no longer be 'w3c.org'; in the case of the Java binding, the package names can no longer be in the 'org.w3c' package.

Note: The original version of the W3C Software Copyright Notice and License could be found at
<http://www.w3.org/Consortium/Legal/copyright-software-19980720>

Copyright 1994-2000 World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved.
<http://www.w3.org/Consortium/Legal/>

This W3C work (including software, documents, or other related items) is being provided by the copyright holders under the following license. By obtaining, using and/or copying this work, you (the licensee) agree that you have read, understood, and will comply with the following terms and conditions:

Permission to use, copy, and modify this software and its documentation, with or without modification, for any purpose and without fee or royalty is hereby granted, provided that you include the following on ALL copies of the software and documentation or portions thereof, including modifications, that you make:

1. The full text of this NOTICE in a location viewable to users of the redistributed or derivative work.

2. Any pre-existing intellectual property disclaimers, notices, or terms and conditions. If none exist, a short notice of the following form (hypertext is preferred, text is permitted) should be used within the body of any redistributed or derivative code: "Copyright <\$date-of-software> World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved.
<http://www.w3.org/Consortium/Legal/>"

3. Notice of any changes or modifications to the W3C files, including the date changes were made. (We recommend you provide URIs to the location from which the code is derived.)

THIS SOFTWARE AND DOCUMENTATION IS PROVIDED "AS IS," AND COPYRIGHT HOLDERS MAKE NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF THE SOFTWARE OR DOCUMENTATION WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.

COPYRIGHT HOLDERS WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF ANY USE OF THE SOFTWARE OR DOCUMENTATION.

The name and trademarks of copyright holders may NOT be used in advertising or publicity pertaining to the software without specific, written prior permission. Title to copyright in this software and any

associated documentation will at all times remain with copyright holders.

International Business Machines Corporation

This product includes software developed by International Business Machines Corporation. Copyright (c) International Business Machines. All rights reserved.

The terms of IBM Public License Version 1.0 apply to the TIBCO End User License Agreement. Any provisions in this License Agreement which differ from the terms of the IBM Public License Version 1.0 are offered by TIBCO Software Inc., alone, and not by any other party. In the event provisions of this License Agreement differ from the terms of the IBM Public License Version 1.0, you may obtain the ICU source code by sending email to support@tibco.com.

"IBM Public License Version 1.0

THE ACCOMPANYING PROGRAM IS PROVIDED UNDER THE TERMS OF THIS IBM PUBLIC LICENSE ("AGREEMENT"). ANY USE, REPRODUCTION OR DISTRIBUTION OF THE PROGRAM CONSTITUTES RECIPIENT'S ACCEPTANCE OF THIS AGREEMENT.

1. DEFINITIONS

"Contribution" means: a) in the case of International Business Machines Corporation ("IBM"), the Original Program, and b) in the case of each Contributor, i) changes to the Program, and ii) additions to the Program; where such changes and/or additions to the Program originate from and are distributed by that particular Contributor. A Contribution 'originates' from a Contributor if it was added to the Program by such Contributor itself or anyone acting on such Contributor's behalf. Contributions do not include additions to the Program which: (i) are separate modules of software distributed in conjunction with the Program under their own license agreement, and (ii) are not derivative works of the Program. "Contributor" means IBM and any other entity that distributes the Program.

"Licensed Patents " mean patent claims licensable by a Contributor which are necessarily infringed by the use or sale of its Contribution alone or when combined with the Program.

"Original Program" means the original version of the software accompanying this Agreement as released by IBM, including source code, object code and documentation, if any.

"Program" means the Original Program and Contributions. "Recipient" means anyone who receives the Program under this Agreement, including all Contributors.

2. GRANT OF RIGHTS

a) Subject to the terms of this Agreement, each Contributor hereby grants Recipient a non-exclusive, worldwide, royalty-free copyright license to reproduce, prepare derivative works of, publicly display, publicly perform, distribute and sublicense the Contribution of such Contributor, if any, and such derivative works, in source code and object code form.

b) Subject to the terms of this Agreement, each Contributor hereby grants Recipient a non-exclusive, worldwide, royalty-free patent license under Licensed Patents to make, use, sell, offer to sell, import and otherwise transfer the Contribution of such Contributor, if any, in source code and object code form. This patent license shall apply to the combination of the Contribution and the Program if, at the time the Contribution is added by the Contributor, such addition of the Contribution causes such combination to be covered by the Licensed Patents. The patent license shall not apply to any other combinations which include the Contribution. No hardware per se is licensed hereunder.

c) Recipient understands that although each Contributor grants the licenses to its Contributions set forth herein, no assurances are provided by any Contributor that the Program does not infringe the

patent or other intellectual property rights of any other entity. Each Contributor disclaims any liability to Recipient for claims brought by any other entity based on infringement of intellectual property rights or otherwise. As a condition to exercising the rights and licenses granted hereunder, each Recipient hereby assumes sole responsibility to secure any other intellectual property rights needed, if any. For example, if a third party patent license is required to allow Recipient to distribute the Program, it is Recipient's responsibility to acquire that license before distributing the Program.

d) Each Contributor represents that to its knowledge it has sufficient copyright rights in its Contribution, if any, to grant the copyright license set forth in this Agreement.

3. REQUIREMENTS A Contributor may choose to distribute the Program in object code form under its own license agreement, provided that: a) it complies with the terms and conditions of this Agreement; and b) its license agreement: i) effectively disclaims on behalf of all Contributors all warranties and conditions, express and implied, including warranties or conditions of title and non-infringement, and implied warranties or conditions of merchantability and fitness for a particular purpose; ii) effectively excludes on behalf of all Contributors all liability for damages, including direct, indirect, special, incidental and consequential damages, such as lost profits; iii) states that any provisions which differ from this Agreement are offered by that Contributor alone and not by any other party; and iv) states that source code for the Program is available from such Contributor, and informs licensees how to obtain it in a reasonable manner on or through a medium customarily used for software exchange. When the Program is made available in source code form: a) it must be made available under this Agreement; and b) a copy of this Agreement must be included with each copy of the Program.

Each Contributor must include the following in a conspicuous location in the Program: Copyright ? {date here}, International Business Machines Corporation and others. All Rights Reserved. In addition, each Contributor must identify itself as the originator of its Contribution, if any, in a manner that reasonably allows subsequent Recipients to identify the originator of the Contribution.

4. COMMERCIAL DISTRIBUTION Commercial distributors of software may accept certain responsibilities with respect to end users, business partners and the like. While this license is intended to facilitate the commercial use of the Program, the Contributor who include the Program in a commercial product offering should do so in a manner which does not create potential liability for other Contributors. Therefore, if a Contributor includes the Program in a commercial product offering, such Contributor ("Commercial Contributor") hereby agrees to defend and indemnify every other Contributor ("Indemnified Contributor") against any losses, damages and costs (collectively "Losses") arising from claims, lawsuits and other legal actions brought by a third party against the Indemnified Contributor to the extent caused by the acts or omissions of such Commercial Contributor in connection with its distribution of the Program in a commercial product offering. The obligations in this section do not apply to any claims or Losses relating to any actual or alleged intellectual property infringement. In order to qualify, an Indemnified Contributor must: a) promptly notify the Commercial Contributor in writing of such claim, and b) allow the Commercial Contributor to control, and cooperate with the Commercial Contributor in, the defense and any related settlement negotiations. The Indemnified Contributor may participate in any such claim at its own expense. For example, a Contributor might include the Program in a commercial product offering, Product X. That Contributor is then a Commercial Contributor. If that Commercial Contributor then makes performance claims, or offers warranties related to Product X, those

performance claims and warranties are such Commercial Contributor's responsibility alone. Under this section, the Commercial Contributor would have to defend claims against the other Contributors related to those performance claims and warranties, and if a court requires any other Contributor to pay any damages as a result, the Commercial Contributor must pay those damages.

5. NO WARRANTY

EXCEPT AS EXPRESSLY SET FORTH IN THIS AGREEMENT, THE PROGRAM IS PROVIDED ON AN "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, EITHER EXPRESS OR IMPLIED INCLUDING, WITHOUT LIMITATION, ANY WARRANTIES OR CONDITIONS OF TITLE, NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Each Recipient is solely responsible for determining the appropriateness of using and distributing the Program and assumes all risks associated with its exercise of rights under this Agreement, including but not limited to the risks and costs of program errors, compliance with applicable laws, damage to or loss of data, programs or equipment, and unavailability or interruption of operations.

6. DISCLAIMER OF LIABILITY

EXCEPT AS EXPRESSLY SET FORTH IN THIS AGREEMENT, NEITHER RECIPIENT NOR ANY CONTRIBUTORS SHALL HAVE ANY LIABILITY FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING WITHOUT LIMITATION LOST PROFITS), HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OR DISTRIBUTION OF THE PROGRAM OR THE EXERCISE OF ANY RIGHTS GRANTED HEREUNDER, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

7. GENERAL

If any provision of this Agreement is invalid or unenforceable under applicable law, it shall not affect the validity or enforceability of the remainder of the terms of this Agreement, and without further action by the parties hereto, such provision shall be reformed to the minimum extent necessary to make such provision valid and enforceable. If Recipient institutes patent litigation against a Contributor with respect to a patent applicable to software (including a cross-claim or counterclaim in a lawsuit), then any patent licenses granted by that Contributor to such Recipient under this Agreement shall terminate as of the date such litigation is filed. In addition, If Recipient institutes patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Program itself (excluding combinations of the Program with other software or hardware) infringes such Recipient's patent(s), then such Recipient's rights granted under Section 2(b) shall terminate as of the date such litigation is filed. All Recipient's rights under this Agreement shall terminate if it fails to comply with any of the material terms or conditions of this Agreement and does not cure such failure in a reasonable period of time after becoming aware of such noncompliance. If all Recipient's rights under this Agreement terminate, Recipient agrees to cease use and distribution of the Program as soon as reasonably practicable. However, Recipient's obligations under this Agreement and any licenses granted by Recipient relating to the Program shall continue and survive. IBM may publish new versions (including revisions) of this Agreement from time to time. Each new version of the Agreement will be given a distinguishing version number.

The Program (including Contributions) may always be distributed subject to the version of the Agreement under which it was received.

In addition, after a new version of the Agreement is published, Contributor may elect to distribute the Program (including its Contributions) under the new version. No one other than IBM has the right to modify this Agreement. Except as expressly stated in Sections 2(a) and 2(b) above, Recipient receives no rights or licenses to the intellectual property of any Contributor under this Agreement, whether expressly, by implication, estoppel or otherwise. All rights in the Program not expressly granted under this Agreement are reserved.

This Agreement is governed by the laws of the State of New York and the intellectual property laws of the United States of America. No party to this Agreement will bring a legal action under this Agreement more than one year after the cause of action arose. Each party waives its rights to a jury trial in any resulting litigation."

Sun Microsystems, Inc. Binary Code License Agreement

READ THE TERMS OF THIS AGREEMENT AND ANY PROVIDED SUPPLEMENTAL LICENSE TERMS (COLLECTIVELY "AGREEMENT") CAREFULLY BEFORE OPENING THE SOFTWARE MEDIA PACKAGE. BY OPENING THE SOFTWARE MEDIA PACKAGE, YOU AGREE TO THE TERMS OF THIS AGREEMENT. IF YOU ARE ACCESSING THE SOFTWARE ELECTRONICALLY, INDICATE YOUR ACCEPTANCE OF THESE TERMS BY SELECTING THE "ACCEPT" BUTTON AT THE END OF THIS AGREEMENT. IF YOU DO NOT AGREE TO ALL THESE TERMS, PROMPTLY RETURN THE UNUSED SOFTWARE TO YOUR PLACE OF PURCHASE FOR A REFUND OR, IF THE SOFTWARE IS ACCESSED ELECTRONICALLY, SELECT THE "DECLINE" BUTTON AT THE END OF THIS AGREEMENT.

1. LICENSE TO USE. Sun grants you a non-exclusive and non-transferable license for the internal use only of the accompanying software and documentation and any error corrections provided by Sun (collectively "Software"), by the number of users and the class of computer hardware for which the corresponding fee has been paid.

2. RESTRICTIONS. Software is confidential and copyrighted. Title to Software and all associated intellectual property rights is retained by Sun and/or its licensors. Except as specifically authorized in any Supplemental License Terms, you may not make copies of Software, other than a single copy of Software for archival purposes. Unless enforcement is prohibited by applicable law, you may not modify, decompile, or reverse engineer Software. You acknowledge that Software is not designed, licensed or intended for use in the design, construction, operation or maintenance of any nuclear facility. Sun disclaims any express or implied warranty of fitness for such uses. No right, title or interest in or to any trademark, service mark, logo or trade name of Sun or its licensors is granted under this Agreement.

3. LIMITED WARRANTY. Sun warrants to you that for a period of ninety (90) days from the date of purchase, as evidenced by a copy of the receipt, the media on which Software is furnished (if any) will be free of defects in materials and workmanship under normal use. Except for the foregoing, Software is provided "AS IS". Your exclusive remedy and Sun's entire liability under this limited warranty will be at Sun's option to replace Software media or refund the fee paid for Software.

4. DISCLAIMER OF WARRANTY. UNLESS SPECIFIED IN THIS AGREEMENT, ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A

PARTICULAR PURPOSE OR NON-INFRINGEMENT ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT THESE DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

5. LIMITATION OF LIABILITY. TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL SUN OR ITS LICENSORS BE LIABLE FOR ANY LOST REVENUE, PROFIT OR DATA, OR FOR SPECIAL, INDIRECT, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER CAUSED REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF OR RELATED TO THE USE OF OR INABILITY TO USE SOFTWARE, EVEN IF SUN HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. In no event will Sun's liability to you, whether in contract, tort (including negligence), or otherwise, exceed the amount paid by you for Software under this Agreement. The foregoing limitations will apply even if the above stated warranty fails of its essential purpose.

6. Termination. This Agreement is effective until terminated. You may terminate this Agreement at any time by destroying all copies of Software. This Agreement will terminate immediately without notice from Sun if you fail to comply with any provision of this Agreement. Upon Termination, you must destroy all copies of Software.

7. Export Regulations. All Software and technical data delivered under this Agreement are subject to US export control laws and may be subject to export or import regulations in other countries. You agree to comply strictly with all such laws and regulations and acknowledge that you have the responsibility to obtain such licenses to export, re-export, or import as may be required after delivery to you.

8. U.S. Government Restricted Rights. If Software is being acquired by or on behalf of the U.S. Government or by a U.S. Government prime contractor or subcontractor (at any tier), then the Government's rights in Software and accompanying documentation will be only as set forth in this Agreement; this is in accordance with 48 CFR 227.7201 through 227.7202-4 (for Department of Defense (DOD) acquisitions) and with 48 CFR 2.101 and 12.212 (for non-DOD acquisitions).

9. Governing Law. Any action related to this Agreement will be governed by California law and controlling U.S. federal law. No choice of law rules of any jurisdiction will apply.

10. Severability. If any provision of this Agreement is held to be unenforceable, this Agreement will remain in effect with the provision omitted, unless omission would frustrate the intent of the parties, in which case this Agreement will immediately terminate.

11. Integration. This Agreement is the entire agreement between you and Sun relating to its subject matter. It supersedes all prior or contemporaneous oral or written communications, proposals, representations and warranties and prevails over any conflicting or additional terms of any quote, order, acknowledgment, or other communication between the parties relating to its subject matter during the term of this Agreement. No modification of this Agreement will be binding, unless in writing and signed by an authorized representative of each party.

JAVA(TM) INTERFACE CLASSES JAVA API FOR XML PROCESSING (JAXP), VERSION 1.1 SUPPLEMENTAL LICENSE TERMS

These supplemental license terms ("Supplemental Terms") add to or modify the terms of the Binary Code License Agreement (collectively, the "Agreement"). Capitalized terms not defined in these Supplemental Terms shall have the same meanings ascribed to them

in the Agreement. These Supplemental Terms shall supersede any inconsistent or conflicting terms in the Agreement, or in any license contained within the Software.

1. **Software Internal Use and Development License Grant.** Subject to the terms and conditions of this Agreement, including, but not limited to Section 3 (Java(TM) Technology Restrictions) of these Supplemental Terms, Sun grants you a non-exclusive, non-transferable, limited license to reproduce internally and use internally the binary form of the Software, complete and unmodified, for the sole purpose of designing, developing and testing your Java applets and applications ("Programs").

2. **License to Distribute Software.** In addition to the license granted in Section 1 (Software Internal Use and Development License Grant) of these Supplemental Terms, subject to the terms and conditions of this Agreement, including but not limited to Section 3 (Java Technology Restrictions), Sun grants you a non-exclusive, non-transferable, limited license to reproduce and distribute the Software in binary form, provided that you (i) distribute the Software complete and unmodified and only bundled as part of your Programs, (ii) do not distribute additional software intended to replace any component(s) of the Software, (iii) do not remove or alter any proprietary legends or notices contained in the Software, (iv) only distribute the Software subject to a license agreement that protects Sun's interests consistent with the terms contained in this Agreement, and (v) agree to defend and indemnify Sun and its licensors from and against any damages, costs, liabilities, settlement amounts and/or expenses (including attorneys' fees) incurred in connection with any claim, lawsuit or action by any third party that arises or results from the use or distribution of any and all Programs and/or Software.

3. **Java Technology Restrictions.** You may not modify the Java Platform Interface ("JPI", identified as classes contained within the "java" package or any subpackages of the "java" package), by creating additional classes within the JPI or otherwise causing the addition to or modification of the classes in the JPI. In the event that you create an additional class and associated API(s) which (i) extends the functionality of the Java platform, and (ii) is exposed to third party software developers for the purpose of developing additional software which invokes such additional API, you must promptly publish broadly an accurate specification for such API for free use by all developers. You may not create, or authorize your licensees to create additional classes, interfaces, or subpackages that are in any way identified as "java", "javax", "sun" or similar convention as specified by Sun in any naming convention designation.

4. **Trademarks and Logos.** You acknowledge and agree as between you and Sun that Sun owns the SUN, SOLARIS, JAVA, JINI, FORTE, STAROFFICE, STARPORTAL and iPLANET trademarks and all SUN, SOLARIS, JAVA, JINI, FORTE, STAROFFICE, STARPORTAL and iPLANET-related trademarks, service marks, logos and other brand designations ("Sun Marks"), and you agree to comply with the Sun Trademark and Logo Usage Requirements currently located at <http://www.sun.com/policies/trademarks>. Any use you make of the Sun Marks inures to Sun's benefit.

5. **Source Code.** Software may contain source code that is provided for reference purposes pursuant to the terms of this Agreement. Source code may not be redistributed unless expressly provided for in this Agreement. Portions of this download are governed by the Apache Source Code License and are identified in the Readme file. A copy of the Apache License is supplied with the Apache Source Code.

6. **Termination for Infringement.** Either party may terminate this Agreement immediately should any Software become, or in either

party's opinion be likely to become, the subject of a claim of infringement of any intellectual property right.

For inquiries please contact: Sun Microsystems, Inc. 901 San Antonio Road, Palo Alto, California 94303

Microstar Software.

Portions of this Software may include software provided by Microstar Software. Copyright (c) 1997, 1998 by Microstar Software. All rights reserved.

Index

A

Active Server Page (ASP) Demonstration
 ASPDemo 161
 adapter
 component (operation) information through TIBCO
 Hawk 210
 adapter instance identifier 136
 agents 198
 alerts 198
 ASPDemo 149

B

BindToObject 140

C

C++ 140
 C++ Demonstration Programs 150
 Class Microagent Name field, adapter 56
 clientvar property 103
 COM data types 186
 command line arguments 211
 configuration properties, retrieving through TIBCO
 Hawk 211
 Configuring adcomService 126
 Configuring Interceptor 126
 customer support xiii

D

DCOM support 123

E

endpoint
 override 137
 ENUMs 186

F

fault tolerance 86

G

General Comments on Data Types 190
 getHostInformation()
 Hawk method 213
 GetObject 139
 global variables 118

H

Hawk methods
 getHostInformation() 213
 preRegisterListener() 218
 High Availability 125
 HTTP 105
 HTTPS 105
 URLfile 107

I

- identifier
 - adapter instance 136
 - programmatic 137
- Interceptor 6, 8, 136, 139
- ITestobj1 151, 152, 157
- ITestobj2 151, 152
- ITestobj3 151, 152

L

- ledger files
 - retrieving information through TIBCO Hawk 219
- load balancing 86
- Load Balancing Using JMS 128
- Load Balancing Using RVCMQ 126
- Log File field, adapter 55
- Log Info field, adapter 55
- Log to Standard field, adapter 55

M

- Mapping COM and TIBCO ActiveEnterprise Data Types 144
- Mapping endpoints and COM Coclasses 141
- Mapping Properties in Repository 142
- method 5, 8, 140, 144, 148, 151, 152, 157, 161
- method and AE exceptions 129
- method call 8
- method signature 141
 - illegal 190
- Method Signatures Must Match 141
- microagent methods supported 204
- Microagent Session field, adapter 57
- MkParseDisplayName 140
- moniker 2, 7, 134, 137, 140
- moniker, BizTalk sked 171

O

- Obtaining Extended Error Information 131
- operation 148, 150
- override endpoint 137
- Overview 148
- Overview of Interceptor Activation 133

P

- password property 101
- preRegisterListener()
 - Hawk method 218
- programmatic identifier 137
- proxy 140

R

- Recommended Deployment Architecture 13
- Request Response Invocation Service 142
- Request Response Service 141
- RequestForm.asp 161
- ResultForm.asp 161
- reviewLedger, TIBCO Hawk method 219

S

- SAFEARRAYs 185
- Service 8
- Standard Microagent Name field, adapter 56
- substitution 118
- Support for Properties 142
- support, contacting xiii

T

- technical support xiii

- testinterceptor 149, 151
- testinterceptorCm 149, 152
- TIBCO Hawk
 - background information 198
 - enterprise monitor components 198
- TIBCO Hawk methods
 - getComponents 210
 - getConfig 211
 - getRvConfig 214
 - getStatus 215
 - reviewLedger 219
- TIBCO Rendezvous 104
- TIBCO Rendezvous, retrieving configuration through
 - TIBCO Hawk 214
- ToUpper 157, 161
- Tracing 232
- Tracing Levels and Fields 234

U

- URLfile accessed via HTTPS 107
- Use Advanced Logging field, adapter 54
- User Defined Types (UDTs) 186
- username property 101
- Using TIBCO Moniker in VC++ 140
- Using TIBCO Moniker in Visual Basic 139

V

- variable substitution 118
- variables 118
- VARIANTs 184
- VBDemo 149
- VBPubSubDemo 149
- Visual Basic 139
- Visual Basic Demonstration Program
 - VBOperationDemo 157

