

TIBCO ActiveMatrix[®] Adapter for Amdocs CRM

Configuration and Deployment

*Software Release 5.5
July 2012*

Important Information

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

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

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.

TIBCO, The Power of Now, TIBCO Rendezvous, TIBCO Runtime Agent, TIBCO Designer, TIBCO Administrator, TIBCO ActiveMatrix BusinessWorks, TIBCO Enterprise Message Service, TIBCO Adapter SDK, and TIBCO Hawk are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

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

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

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

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

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

Copyright © 1999-2012 TIBCO Software Inc. ALL RIGHTS RESERVED.

TIBCO Software Inc. Confidential Information

Contents

Figures	ix
Tables	xi
Preface	xv
Changes from the Previous Release of this Guide	xvi
Related Documentation	xvii
TIBCO ActiveMatrix Adapter for Amdocs CRM Documentation	xvii
Other TIBCO Product Documentation	xvii
Typographical Conventions	xviii
Connecting with TIBCO Resources	xx
How to Join TIBCOCommunity	xx
How to Access TIBCO Documentation	xx
How to Contact TIBCO Support	xx
Chapter 1 Overview of the Adapter	1
Introduction to the Adapter	2
Adapter Features	3
Chapter 2 Getting Started	5
Overview of Getting Started	6
Prerequisites	7
Configuring the Adapter	8
Create a Project	8
Configure an Amdocs Adapter Instance	9
Add and Configure the Adapter Service	11
Starting the Adapter	14
Chapter 3 Adapter Instance Options	15
Overview of Adapter Instance Options	16
Start the Design-time Adapter	16
Configuration Tasks	16
Adapter Instance Tabs	18
Configuration Tab	18

Design-time Connection Tab	20
Run-time Connection Tab	21
Adapter Services Tab	22
General Tab	24
Logging Tab	25
Startup Tab	29
Monitoring Tab	29
Chapter 4 Adapter Services Options	31
Overview of Adapter Services Options	32
Setting Schema	32
Message Transports	32
Publication Service	36
Configuration Tab	36
Fetch CBO Tab	37
Transport Tab	37
Schema Tab	40
Subscription Service	41
Configuration Tab	41
Fetch CBO Tab	42
Transport Tab	42
Schema Tab	44
Request-Response Service	45
Configuration Tab	45
Fetch CBO Tab	46
Transport Tab	46
Schema Tab	48
Request-Response Invocation Service	49
Configuration Tab	49
Fetch Request CBO Tab	50
Fetch Reply CBO Tab	50
SubscriberOptions Tab	51
Transport Tab	51
Schema Tab	53
Chapter 5 Publication Service Functionality	55
Publication Service Overview	56
Publication Service Features	56
How the Publication Service Works	60
Preparing Amdocs CRM Components for Publication Service	61
Add TIBCO Staging Tables to Amdocs CRM Database	61
Enable Publication Service for LAN Client	62

Enable the Publication Service for Amdocs CRM Web Client	69
Enable the Publication Service for Amdocs CRM Applications	74
Limitations of the Publication Service	82
Chapter 6 Subscription Service Functionality	83
Subscription Service Overview	84
Subscription Service Features	84
Subscription Service Functionality	84
How the Subscription Service Works	86
CBO Wrapper Invocation	86
Custom JavaBean Invocation	87
Service Error Handling	91
Chapter 7 Request-Response Service Functionality	93
Request-Response Service Overview	94
Request-Response Service Features	94
Supported Request-Response Operations	94
How the Request-Response Service Works	95
Query Operations	95
Workflow Operations	97
Customized Operations	100
Custom JavaBean Invocation	102
Chapter 8 Request-Response Invocation Service Functionality	103
Request-Response Invocation Service Overview	104
Request-Response Invocation Service Using Flexible Deployment	104
Request-Response Invocation Service Using Non-Flexible Deployment	105
How the Request-Response Invocation Service Works	106
ClearBasic Forms	106
Web Forms	114
Chapter 9 Deploying and Starting an Adapter Using TIBCO Administrator	119
Creating an EAR File in TIBCO Designer	120
Deploying the Project	121
Starting or Stopping the Adapter	123
Monitoring the Adapter	124
Chapter 10 Configuring Advanced Features	125
Using the Adapter with a Revision Control System	126
Modifying the DTA Rendezvous Connection Properties	128

Defining a TIBCO Hawk Session	129
Using Global Variables	130
Specifying Global Variables Using TIBCO Designer	130
Using Global Variables in the Project	130
Changing Global Variable Values at Runtime	132
Predefined Global Variables	133
Configuring a Remote Adapter	136
Using the Tracking Element	137
Chapter 11 Monitoring the Adapter Using TIBCO Hawk	139
Overview	140
Starting TIBCO Hawk Software	141
The Auto-Discovery Process	142
Invoking Microagent Methods	144
Available Microagents	147
activateTraceRole()	150
deactivateTraceRole()	151
getActivityStatistics()	152
getActivityStatisticsByOperation(Operation)	153
getActivityStatisticsBySchema(SchemaName)	154
getActivityStatisticsByService(ServiceName)	155
getAdapterServiceInformation()	156
getComponents()	157
getConfig()	158
getConfigProperties()	159
getConnectionStatistics()	160
getHostInformation()	161
getPerfMonSetting()	162
getPollingBatchSize()	163
getPollingInterval()	164
getQueueStatistics()	165
getRvConfig()	166
getStatus()	167
getThreadStatistics()	168
getTraceSinks()	169
getVersion()	170
_onUnsolicitedMsg()	171
preRegisterListener()	172
resetActivityStatistics()	173
resetConnectionStatistics()	174
resetThreadStatistics()	175
reviewLedger()	176

setPollingBatchSize()	177
setPollingInterval()	178
setTraceSinks()	179
stopApplicationInstance()	180
unRegisterListener()	181
Appendix A TIBCO Staging Tables	183
table_tibco_message_queue	184
table_tibco_message_fields	185
Appendix B Error Messages	189
Error Message Listing	190
Appendix C User Exits	231
Overview	232
How the Adapter Implements User Exits	233
IUserExit Interface	234
Appendix D Wrapper Classes	237
Overview	238
Wrappers Packaged with the Adapter	239
ActEntry_Wrapper	239
Address_Wrapper	239
BusOrg_Wrapper	239
Case_Wrapper	239
Communication_Wrapper	240
Contact_Wrapper	240
CreditCard_Wrapper	240
DemandDetail_Wrapper	240
DemandHeader_Wrapper	241
Dialogue_Wrapper	241
DocInst_Wrapper	242
Generic_Edr_Com_Role Wrapper	242
NotesLog_Wrapper	242
OEQuote_Wrapper	242
PartPrice_Wrapper	243
Part_Wrapper	243
ServicePart_Wrapper	244
ShopList_Wrapper	244
Site_Wrapper	244
SubCase_Wrapper	244
Task_Wrapper	245

User_Wrapper 245

Wrapper Class and TibcoCBOInterface 246

Writing Wrapper Class for Case Business Object. 247

 Steps Involved 247

Appendix E Testing the Adapter 257

Testing the Publication Service with ClearBasic Forms 258

 Importing ClearBasic Files. 258

Testing the Publication Service with Web Forms 262

Testing the Request-Response Functionality with Web Forms. 264

Testing the Request-Response Functionality with ClearBasic Forms. 265

Appendix F Frequently Asked Questions 267

Frequently Asked Questions. 268

Index 271

Figures

Figure 1	TIBCO Designer startup panel	8
Figure 2	TIBCO Designer Configuration tab	9
Figure 3	Design-time Connection settings	10
Figure 4	Run-time Connection settings	11
Figure 5	Adapter Services folder	12
Figure 6	Add Publication Service	12
Figure 7	Change the polling interval.	24
Figure 8	Generate sinks	28
Figure 9	Directive File window	63
Figure 10	Event Configuration	75
Figure 11	Select Trigger Object	76
Figure 12	Trigger Configuration	77
Figure 13	Add the fields to be logged.	78
Figure 14	Associate the Publisher with the event	79
Figure 15	Amdocs System Configurator	80
Figure 16	Flexible Deployment Mode.	107
Figure 17	Non-Flexible Deployment Mode.	110
Figure 18	Web client deployment mode.	115
Figure 19	Hawk session	129
Figure 20	Global variables	132
Figure 21	TIBCO Hawk Enterprise Monitor	142
Figure 22	Select a method in the Microagents, Methods and Arguments dialog	145
Figure 23	Invocation results dialog.	146
Figure 24	Add Users to Resource Configuration	259
Figure 25	Create Customized Form	260
Figure 26	Directive window	261

Tables

Table 1	General Typographical Conventions	xviii
Table 2	Adapter Instance Configuration Tab	18
Table 3	Adapter Instance Design-time Connection Tab	20
Table 4	Adapter Instance Run-time Connection Tab	21
Table 5	Adapter Instance Adapter Services Tab	23
Table 6	Adapter Instance General Tab	24
Table 7	Adapter Instance Logging Tab	26
Table 8	Adapter Instance Startup Tab	29
Table 9	Adapter Instance Monitoring Tab	30
Table 10	Publication Service: Configuration Tab	36
Table 11	Publication Service: Transport Tab	38
Table 12	Publication Service: Schema Tab	40
Table 13	Subscription Service: Configuration Tab	41
Table 14	Subscription Service: Transport Tab	42
Table 15	Subscription Service: Schema Tab	44
Table 16	Request-Response Service: Configuration Tab	45
Table 17	Request-Response Service: Transport Tab	46
Table 18	Request-Response Service: Schema Tab	48
Table 19	Request-Response Invocation Service: Configuration Tab	49
Table 20	Request-Response Invocation Service: SubscriberOptions Tab	51
Table 21	Request-Response Invocation Service: Transport Tab	51
Table 22	Request-Response Invocation Service: Schema Tab	53
Table 23	Incoming Event	57
Table 24	OpObject	58
Table 25	Field	58
Table 26	Relation	59
Table 27	Message Formats that Contain CBOs	59
Table 28	ClearBasic Functions	64

Table 29	Java Methods	69
Table 30	Attributes of the incomingEvent Class	85
Table 31	Relevant Fields for the Custom JavaBean Functionality	85
Table 32	CBOs Provided with Wrapper Classes	86
Table 33	Incoming Message Formats for Query Operations	96
Table 34	Outgoing Message Formats for Query Operations	97
Table 35	Incoming Message Formats for Workflow Operations	99
Table 36	Outgoing Message Formats for Workflow Operations	100
Table 37	Incoming Message Formats for Customized Operations	101
Table 38	Outgoing Message Formats for Customized Operations	101
Table 39	Parameter List of Function TibcoSetMessageFields()	116
Table 40	Parameter list of function TibcoSetMessageFieldsEMS()	117
Table 41	Predefined Global Variables	133
Table 42	Microagent Methods	147
Table 43	activateTraceRole()	150
Table 44	deactivateTraceRole()	151
Table 45	Input parameter of getActivityStatistics()	152
Table 46	Returns of getActivityStatistics()	152
Table 47	Input parameter of getActivityStatisticsByOperation(Operation)	153
Table 48	Returns of getActivityStatisticsByOperation(Operation)	153
Table 49	Input parameter of getActivityStatisticsBySchema(SchemaName)	154
Table 50	Returns of getActivityStatisticsBySchema(SchemaName)	154
Table 51	Input Parameter of getActivityStatisticsByService(ServiceName)	155
Table 52	Returns of getActivityStatisticsByService(ServiceName)	155
Table 53	Input Parameter of getAdapterServiceInformation()	156
Table 54	Returns of getAdapterServiceInformation()	156
Table 55	Input Parameters of getComponents()	157
Table 56	Returns of getComponents()	157
Table 57	getConfig()	158
Table 58	Input Parameter of getConfigProperties()	159
Table 59	Returns of getConfigProperties()	159
Table 60	getConnectionStatistics()	160

Table 61	getHostInformation()	161
Table 62	getPerfMonSetting()	162
Table 63	Input Parameter of getPollingBatchSize()	163
Table 64	Returns of getPollingBatchSize()	163
Table 65	getPollingInterval()	164
Table 66	getQueueStatistics()	165
Table 67	Input Parameter of getRvConfig()	166
Table 68	Returns of getRvConfig()	166
Table 69	getStatus()	167
Table 70	getThreadStatistics()	168
Table 71	Input Parameters of getTraceSinks()	169
Table 72	Returns of getTraceSinks()	169
Table 73	getVersion()	170
Table 74	preRegisterListener()	172
Table 75	Input Parameters of reviewLedger()	176
Table 76	Returns of reviewLedger()	176
Table 77	setPollingBatchSize()	177
Table 78	setPollingInterval()	178
Table 79	setTraceSinks()	179
Table 80	unRegisterListener()	181
Table 81	table_tibco_message_queue	184
Table 82	table_tibco_message_fields	185
Table 83	Error Message	190

Preface

TIBCO ActiveMatrix Adapter for Amdocs CRM serves as a bi-directional gateway between a Amdocs CRM system and applications configured for the TIBCO environment.

Topics

- [Changes from the Previous Release of this Guide, page xvi](#)
- [Related Documentation, page xvii](#)
- [Typographical Conventions, page xviii](#)
- [Connecting with TIBCO Resources, page xx](#)

Changes from the Previous Release of this Guide

This section itemizes the major changes from the previous release of this guide.

Renaming TIBCO Adapter for ClarifyCRM

The adapter is renamed to TIBCO ActiveMatrix Adapter for Amdocs CRM.

Separating from TIBCO Adapter for ClarifyCRM User's Guide

TIBCO ActiveMatrix Adapter for Amdocs CRM Configuration and Deployment is separated from TIBCO Adapter for ClarifyCRM User's Guide version 5.4.0. It includes most of the chapters in the User's Guide except for the Concepts, Installation, and Examples chapters.

Adding Overview of Adapter

A chapter is added to introduce adapter features, configuration, and deployment. See [Chapter 1, Overview of the Adapter, on page 1](#) for more information.

Adding Introduction to Using TIBCO Administrator

A chapter is added to introduce how to deploy and start an adapter using TIBCO Administrator. See [Chapter 9, Deploying and Starting an Adapter Using TIBCO Administrator, on page 119](#) for more information.

Appending tibco.sch File Directly When Adding TIBCO Staging Tables

Appending the tibco.sch file to the generated schema file directly is supported when adding TIBCO Staging tables to the Amdocs CRM database. The process that converts tibco.sch to an XML file is omitted. See [Add TIBCO Staging Tables to Amdocs CRM Database on page 61](#) for more information.

Adding the Configuration of ClarifyEnv for Event Processor

Configuring ClarifyEnv in Amdocs System Configurator is added before running Event Processor. See [Setting Up Event Processor on page 79](#) for more information.

Related Documentation

This section lists documentation resources you may find useful.

TIBCO ActiveMatrix Adapter for Amdocs CRM Documentation

The following documents form the TIBCO ActiveMatrix Adapter for Amdocs CRM documentation set:

- *TIBCO ActiveMatrix Adapter for Amdocs CRM Concepts* Read this manual to gain an understanding of the product that you can apply to the various tasks you may undertake.
- *TIBCO ActiveMatrix Adapter for Amdocs CRM Installation* Read this manual for instructions on site preparation and installation.
- *TIBCO ActiveMatrix Adapter for Amdocs CRM Configuration and Deployment* Read this manual for instructions on creating, configuring, and deploying adapter projects.
- *TIBCO ActiveMatrix Adapter for Amdocs CRM Examples* Read this manual to work through the examples provided with the adapter.
- *TIBCO ActiveMatrix Adapter for Amdocs CRM Release Notes* Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

Other TIBCO Product Documentation

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

- TIBCO Designer™
- TIBCO Administrator™
- TIBCO ActiveMatrix BusinessWorks™
- TIBCO Rendezvous®
- TIBCO Enterprise Message Service™
- TIBCO Hawk®
- TIBCO® Adapter SDK
- TIBCO Runtime Agent™




Typographical Conventions

The following typographical conventions are used in this manual.

Table 1 General Typographical Conventions

Convention	Use
<i>TIBCO_HOME</i> <i>ENV_NAME</i> <i>AMDOCSCRM_HOME</i>	<p>TIBCO products are installed into an installation environment. A product installed into an installation environment does not access components in other installation environments. Incompatible products and multiple instances of the same product must be installed into different installation environments.</p> <p>An installation environment consists of the following properties:</p> <ul style="list-style-type: none">• Name Identifies the installation environment. This name is referenced in documentation as <i>ENV_NAME</i>. On Microsoft Windows, the name is appended to the name of Windows services created by the installer and is a component of the path to the product shortcut in the Windows Start > All Programs menu.• Path The folder into which the product is installed. This folder is referenced in documentation as <i>TIBCO_HOME</i>. <p><i>AMDOCSCRM_HOME</i> is the directory where the Amdocs CRM Application server is installed. For example, if the Amdocs CRM Application server is installed at C:\Amdocs8.1 on Windows systems, the value of <i>AMDOCSCRM_HOME</i> is C:\Amdocs8.1.</p>
code font	<p>Code font identifies commands, code examples, filenames, pathnames, and output displayed in a command window. For example:</p> <p>Use MyCommand to start the foo process.</p>
bold code font	<p>Bold code font is used in the following ways:</p> <ul style="list-style-type: none">• In procedures, to indicate what a user types. For example: Type admin.• In large code samples, to indicate the parts of the sample that are of particular interest.• In command syntax, to indicate the default parameter for a command. For example, if no parameter is specified, MyCommand is enabled: MyCommand [enable disable]

Table 1 General Typographical Conventions (Cont'd)

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

Connecting with TIBCO Resources

How to Join TIBCOCommunity

TIBCOCommunity is an online destination for TIBCO customers, partners, and resident experts; a place to share and access the collective experience of the TIBCO community. TIBCOCommunity offers forums, blogs, and access to a variety of resources. To register, go to <http://www.tibcommunity.com>.

How to Access TIBCO Documentation

You can access TIBCO documentation here:

<http://docs.tibco.com>

How to Contact TIBCO Support

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

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

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

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

<https://support.tibco.com>

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

Chapter 1 **Overview of the Adapter**

This chapter provides information on the TIBCO ActiveMatrix Adapter for Amdocs CRM features.

Topics

- [Introduction to the Adapter, page 2](#)
- [Adapter Features, page 3](#)

Introduction to the Adapter

TIBCO ActiveMatrix Adapter for Amdocs CRM provides seamless connectivity from the TIBCO environment to the Amdocs CRM system, allows events to be propagated from the Amdocs CRM system to the TIBCO environment and vice versa.

The adapter exports data from and imports data into a Amdocs CRM system in an event-driven fashion. Data is sent out to the TIBCO environment when changes are made in the Amdocs CRM System. When the adapter receives a message containing data to import, the adapter parses the data and inserts it into the Amdocs CRM database.

Adapter Features

This section summarizes the features of the adapter:

- **Support for Multithreading**—The adapter supports a static number of threads. The number of threads is specified at the time of configuration. The number of connections to the application server equals the number of threads. With multiple threads, the adapter processes multiple messages concurrently.
- **Support for Amdocs CRM Business Objects (CBOs)**—The adapter supports CBOs, which are Java Beans and ActiveX controls that provide access to data in the CeFO database.
- **Support for Invoking Custom JavaBean**—The Subscription Service can be used to invoke a Custom JavaBean method. See [Custom JavaBean Invocation on page 83](#) for more information.
- **Support for Parent-child Schemas in design time**—The Fetch CBO tab can be used to fetch both parent CBOs and child CBOs and it displays the parent-child representation. The same representation is displayed under the AESchemas folder.
- **Support for Insert and Update Operations**—The adapter supports INSERT and UPDATE operations for the Request-Response and Subscription services on Existing, Customized and Extended CBOs.
- **Support for Fetching Multiple Rows by Query Operation in Request-Response Service**—The Request-Response Service now supports QUERY operation resulting in multiple rows of the main CBO.
- **Support for Simple Single CBO Query Operation in Request-Response Service**—The Request-Response Service provides support for simple single CBO QUERY operation.
- **Support for Classic-LAN and Web Client**—The adapter provides features for working with both Amdocs CRM Classic-LAN and Web Clients.
- **Support for Workflow Operations**—Amdocs CRM provides a set of workflow operations to make it easier for the user to create and update the workflow objects (case, OEQuote, and so on). Workflow objects also called queue-able objects in the Amdocs CRM system involves the most complicated business logic when workflow objects are created or changed. See [Workflow Operations on page 93](#) for more information.
- **Support for JMS at design time**—Communication between the palette and the design-time adapter is now possible using the JMS transport in addition to the TIBCO Rendezvous transport.
- **Atomicity**—Data is not imported or exported unless the entire business event successfully passes business logic and validation from the Amdocs CRM system.

- **Support for Delayed Acknowledgement**—The adapter ensures transaction integrity by supporting delayed acknowledgement. The adapter sends the acknowledgement to the source of the message received by the Subscription Service only after the message has been consumed in the Amdocs CRM system.
- **A Convenient Configuration Tool**—The adapter is configured using TIBCO Designer. TIBCO Designer connects to a specified Amdocs CRM system, downloads the CBOs for the required services, and transfers the schema and message routing rules for individual services to a central repository. Support for usage of global and client variables ensures easy and quick migration of adapter configurations from development to testing, and from testing to production environments. See *TIBCO ActiveMatrix Adapter for Amdocs CRM Concepts* for more information.
- **Support for TIBCO ActiveMatrix BusinessWorks**—The adapter can be used in a TIBCO ActiveMatrix BusinessWorks process.
- **Exception Handling, Monitoring and Message Tracking**—The adapter employs effective exception-handling techniques and extensive audit trails. The adapter can be configured to work in concert with the TIBCO Hawk monitoring component to detect and handle exception situations. Each message that is processed by an adapter service is tagged with a tracking-ID that enables complete end-to-end tracking of messages in the TIBCO environment.
- **Security**—The Obfuscation engine packaged with the adapter enables encryption of sensitive configuration information.

Chapter 2 **Getting Started**

This chapter describes the basic steps required to configure an adapter service and run the adapter.

Topics

- [Overview of Getting Started, page 6](#)
- [Prerequisites, page 7](#)
- [Configuring the Adapter, page 8](#)
- [Starting the Adapter, page 14](#)

Overview of Getting Started

This chapter gives you a hands-on experience on the various steps involved in configuring adapter. This involves the following steps:

1. [Configuring the Adapter, page 8](#)
2. [Starting the Adapter, page 14](#)

Since the adapter and Amdocs CRM are closely integrated, there are some prerequisites to be met on the Amdocs CRM side before you can start TIBCO Designer to configure even minimum settings for the adapter. See [Prerequisites on page 7](#) for details.

Prerequisites

Before starting the configuration, you need to do the following:

- Ensure that the design-time adapter and the Database Server are running before you start configuration.
- Prepare Amdocs CRM components for publishing which is demonstrated in the example in this chapter. See [Preparing Amdocs CRM Components for Publication Service on page 59](#) for details.
- Be familiar with TIBCO Designer. TIBCO Designer documentation can be accessed by selecting the **Help > Designer Help** menu option in TIBCO Designer.

Install Software

- Install all required software (see *TIBCO ActiveMatrix Adapter for Amdocs CRM Installation*).
- Install the adapter software (see *TIBCO ActiveMatrix Adapter for Amdocs CRM Installation*).

Information You Need

Before starting to configure the adapter, you should have the following information ready:

- Name of the computer hosting the database server
- User ID and password for logging into the Amdocs CRM system
- Name of the Database server

Configuring the Adapter

Each project contains one or more instances of the adapter configuration. This configuration is accessed whenever an adapter application is started.

A typical sequence of creating a project and configuring an adapter instance and related services is as follows:

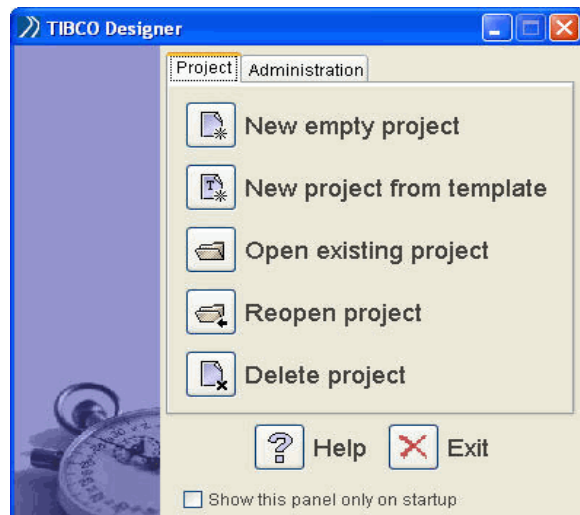
1. [Create a Project, page 8](#)
2. [Configure an Amdocs Adapter Instance, page 9](#)
3. [Add and Configure the Adapter Service, page 11](#)

Create a Project

To create a project in TIBCO Designer, follow these steps:

1. Start TIBCO Designer.
2. Click the **New empty project** button in the TIBCO Designer dialog.

Figure 1 TIBCO Designer startup panel



3. Select the **Multi-File Project** tab in the Save Project dialog. Click the **Browse** button to select the location of project or type the path in the Project Directory field.

For example, type `D:\adclerm` in the Project Directory field.

4. Click the **OK** button.

Configure an Amdocs Adapter Instance

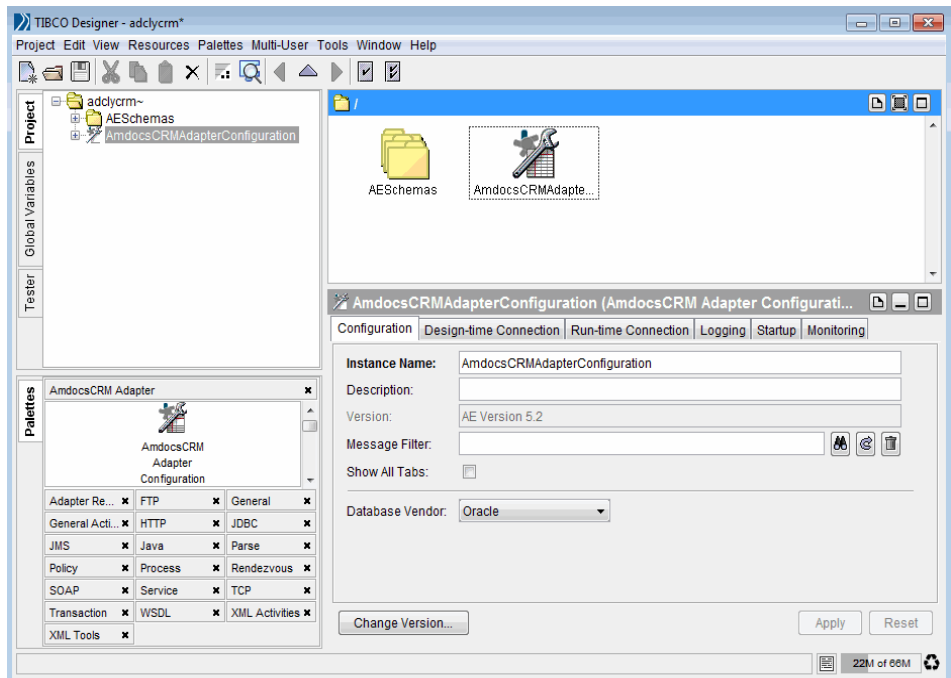
To create and configure an Amdocs adapter instance, follow these steps:

1. Click to open the newly created project in the Project panel.
2. Drag the **AmdocsCRMAdapterConfiguration** icon from the Palette panel into the Design panel.
3. Select the **AmdocsCRMAdapterConfiguration** icon in the Design panel. The AmdocsCRMAdapter Configuration panel appears.

In the Configuration tab:

- a. Specify the instance name in the Instance Name field.
- b. Check the **Show All Tabs** checkbox.
- c. Click the **Apply** button.

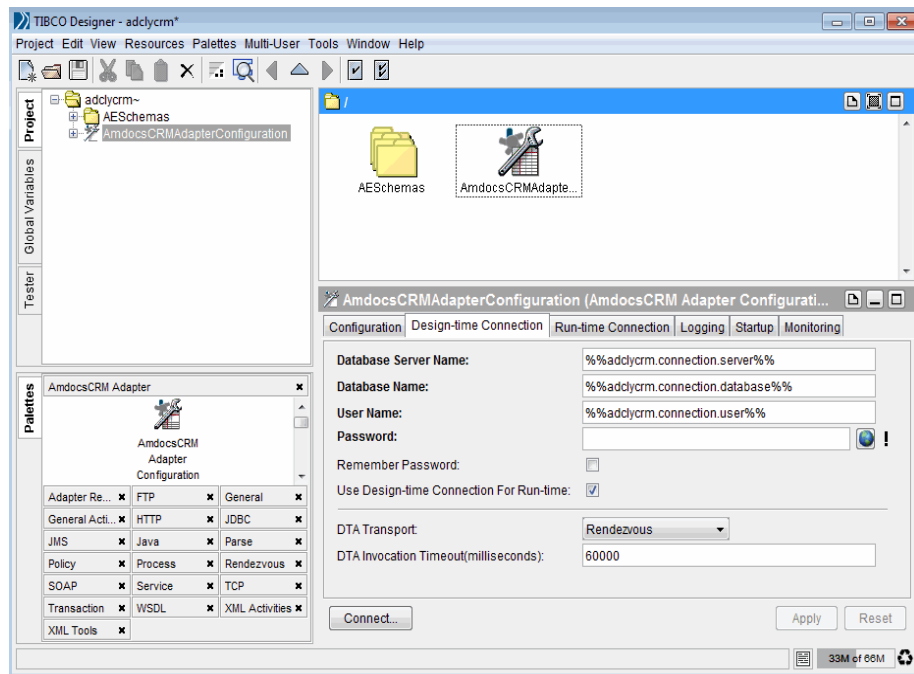
Figure 2 TIBCO Designer Configuration tab



In the Design-time Connection tab:

- Enter the information to connect the Amdocs Application Server in the fields which are marked in bold.
- Select the transport to be used by the design-time adapter in the DTA Transport drop-down list, JMS or Rendezvous.
- Click the **Connect...** button to establish the connection to the design-time adapter.
- Leave other default values unchanged and click the **Apply** button.

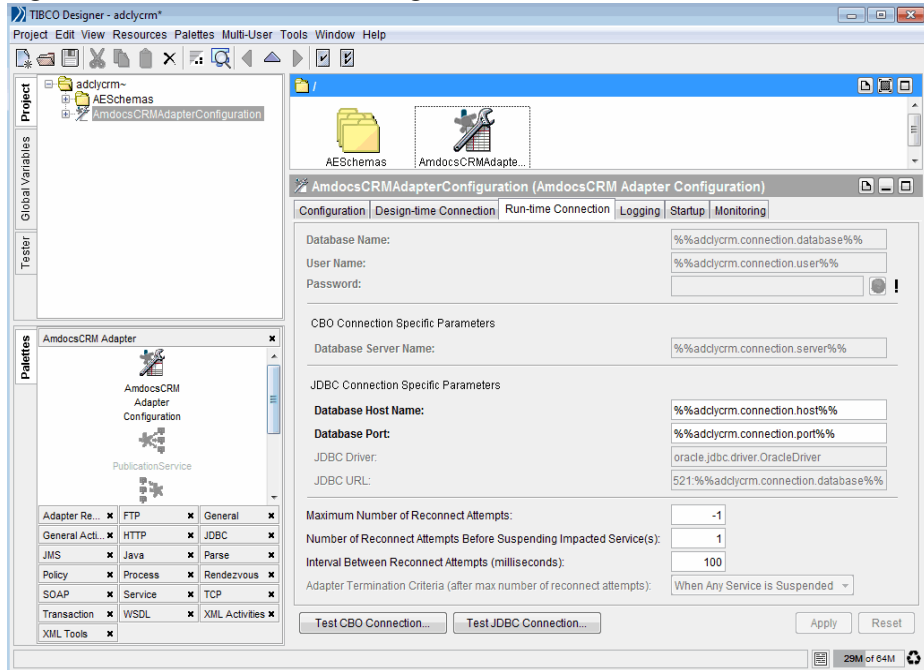
Figure 3 Design-time Connection settings



In the Run-time Connection tab:

- Enter the Database host name and Database port, as shown in [Figure 4](#).
- Leave all other settings unchanged and click the **Apply** button.

Figure 4 Run-time Connection settings



If you select Oracle as the Database Vendor in the Configuration tab, the Database Port field should be 1521. If you select Microsoft SQL Server, the Database Port field should be 1433.

Add and Configure the Adapter Service

After configuring an Amdocs CRM adapter instance, add and configure the adapter services in the adapter.

The following four services can be added:

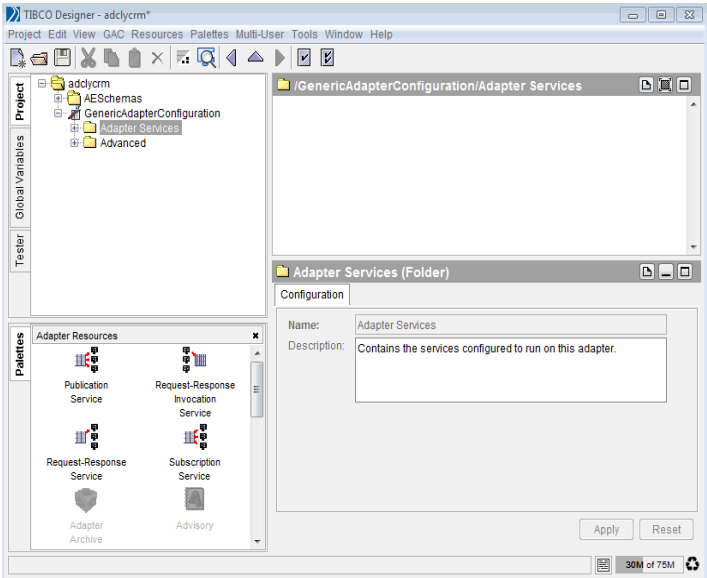
- [Publication Service, page 36](#)
- [Subscription Service, page 40](#)
- [Request-Response Service, page 44](#)
- [Request-Response Invocation Service, page 48](#)

In this chapter, take the Publication Service for example to introduce how to add and configure an adapter service.

To add and configure the Publication Service in the adapter, follow these steps:

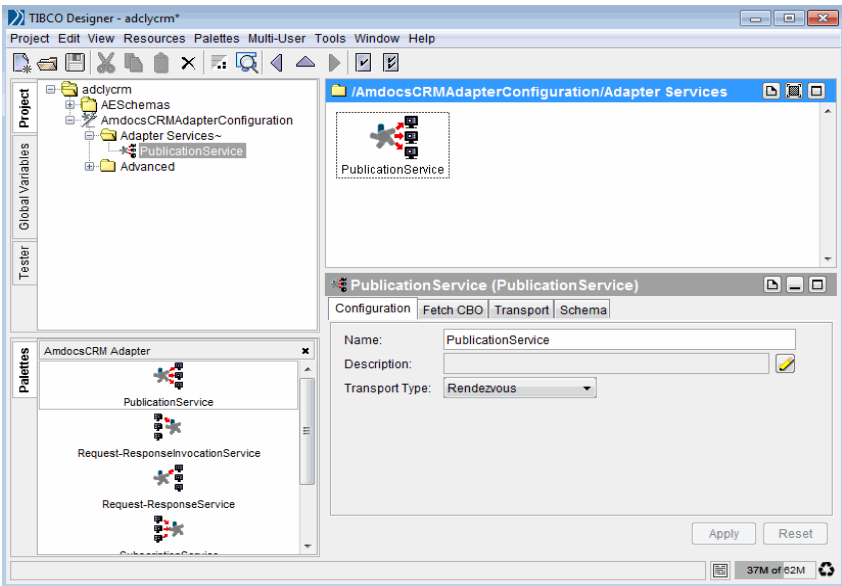
1. Double-click the **Adapter Services** folder in the Project panel.

Figure 5 Adapter Services folder



2. Drag the **PublicationService** icon from the Palette panel to the Design panel. The Publication Service pane appears.

Figure 6 Add Publication Service



3. Click the **Fetch CBO** tab to fetch the CBO.
 - a. Fetch the Main CBO. You can do either of the following two ways:
 - Enter the CBO name in text field and click the **Fetch CBO** button.
 - Click the **Fetch CBO List...** button and select the Main CBO in the list displayed. Click the **OK** button.
 - b. Click the **Fetch Contained CBO...** button to fetch the Contained CBO from the list.
 - c. Click the **Apply** button.



If you want to remove any of the schemas, select that schema and click the **Remove CBO...** button.

4. In the Transport tab, enter the appropriate subject in the Message Subject field. Select the appropriate quality of service and wire formats in the corresponding fields. And then click the **Apply** button.
5. Save the project.

Starting the Adapter

After configuring the adapter with Publication, Subscription, Request-Response or Request-Response services, you can start the adapter from the command line with a repository file.

To start the adapter from the command line with a repository file, the project must be run as a local repository and saved in DAT (repository) format.

Task A Convert the Project to a Repository File

1. Start TIBCO Designer.
2. Select **Project > Export Full Project...** from the menu. The Export Project dialog appears.
3. Click the **Local Repository** tab, enter the project name and the output directory in the Export Project dialog. Click the **OK** button.
4. In the Create Project dialog, select File Type and TIBCO Messaging Encoding. Click the **Yes** button.

Task B Start the Adapter

1. Specify the path of the DAT file in the repourl in the `adclyCRM.tra` file located in the `TIBCO_HOME\adapter\adclycrm\version_num\bin` directory.
2. In the command window, go to the `TIBCO_HOME\adapter\adclycrm\version_num\bin` directory and run the following command to start the adapter:

```
adclycrm --run --propFile adclyCRM.tra
```

Chapter 3 **Adapter Instance Options**

This chapter explains how to create an adapter instance. All configuration tasks are performed in TIBCO Designer and the information is stored in a project that is later used by the runtime adapter.

Topics

- [Overview of Adapter Instance Options, page 16](#)
- [Adapter Instance Tabs, page 18](#)

Overview of Adapter Instance Options

The adapter can be configured with just a few steps before starting it. Alternatively, you can specify sessions and set configuration parameters to suit your needs before starting the adapter.

All adapter configuration is done using TIBCO Designer. Make sure it has been installed appropriately before continuing. See *TIBCO Designer User's Guide* for more information.

A set of preconfigured projects involving the frequently used Amdocs CRM Business Objects is shipped along with the adapter. You can configure them according to your requirements or start a new configuration. All the adapter configuration information is saved as a project.

Start the Design-time Adapter

Before starting TIBCO Designer, start the design-time adapter if the input files are on a remote system that is not mapped to or mounted on your local system.

- On UNIX systems, start the design-time adapter in the command window.
- On Windows systems, the design-time adapter is installed as a service and starts automatically.

Configuration Tasks

Use the following sequence to create and configure an adapter service.

1. Start TIBCO Designer and open a multi-file project. See *TIBCO Designer User's Guide* for details.
2. Drag the **AdapterConfiguration** icon from the palette panel to the design panel. This creates an adapter instance named `AdapterConfiguration` by default.
3. Define the adapter instance by assigning a new name and optionally change logging options. See [Adapter Instance Tabs, page 18](#) for details.
4. Add one or more services to the adapter instance by dragging a service icon from the palettes panel and dropping it in the design panel.
5. Configure the adapter services for the adapter instance. See [Chapter 4, Adapter Services Options, on page 31](#) for more information.
6. Set the combination of options required for your service at the service level.
7. Save the project and exit TIBCO Designer.

After configuring the adapter, you must create the runtime adapter property file and add the project name and adapter instance name.

Adapter Instance Tabs

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

- [Configuration Tab, page 18](#)
- [Design-time Connection Tab, page 20](#)
- [Run-time Connection Tab, page 21](#)
- [Adapter Services Tab, page 22](#)
- [General Tab, page 24](#)
- [Logging Tab, page 25](#)
- [Startup Tab, page 28](#)
- [Monitoring Tab, page 28](#)

Configuration Tab

After you drag a **AmdocsCRMAdapterConfiguration** icon from the palette panel to the design panel, the Configuration tab is selected by default.

Click **Apply** to apply the changes before leaving this tab.

Table 2 Adapter Instance Configuration Tab

Field	Description
Instance Name	<p>The name of the Adapter Service Engine instance.</p> <p>Use the default name or replace it with a name of your choice.</p> <p>See Guidelines for Choosing an Instance Name on page 19 for more information.</p>
Description	<p>(Optional) A short description of the adapter configuration.</p>
Version	<p>The version string indicates the ActiveEnterprise (AE) format in which the adapter instance is saved. When a new adapter instance is created in TIBCO Designer, the version string is set to AE Version 5.2.</p> <p>To change versions, click the Change Version... button.</p>

Table 2 Adapter Instance Configuration Tab (Cont'd)

Field	Description
Message Filter	<p>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.</p> <p>Filters can be written using TIBCO Adapter SDK. See <i>TIBCO Adapter SDK Programmer's Guide</i> for details.</p>
Show All Tabs	Check this checkbox to display the Adapter Services tab and General tab for configuring advanced options.
Database Vendor	Select the database type to which the adapter connects. You can either select Oracle or Microsoft SQL Server from the drop-down list.

Guidelines for Choosing an 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.
- 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 the next time you rename the adapter instance.

Design-time Connection 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.

Table 3 Adapter Instance Design-time Connection Tab

Field	Description
Database Server Name	<p>If you selected Microsoft SQL Server in the Configuration tab, then specify the name of the server machine.</p> <p>If you selected Oracle in the Configuration tab, then specify the SQL*NET connection string to the Oracle instance.</p>
Database Name	Specify the name of the database you wish to connect to.
User Name	Specify the user name for the account used by the design-time adapter to access the Amdocs CRM system.
Password	Specify the password for the account used by the design-time adapter to access the Amdocs CRM system.
Remember Password	If this checkbox is checked, the password must be entered each time the project is opened. If checked, the password will be stored in the project.
Use Design-time Connection for Run-time	<p>Select this checkbox to change the default design-time connection settings.</p> <p>On checking this checkbox, connection fields are enabled in the Run-time Connection tab. Then you can connect to a different database by entering valid values in the Database Name, User Name, Password, and Database Server Name fields.</p>
DTA Transport	Select the transport to be used by the design-time adapter, JMS or Rendezvous. After selecting the transport, the transport-specific configuration fields display.
DTA Invocation Timeout (milliseconds)	The default value is 60000 milliseconds (60 seconds). If the design-time adapter is unable to connect to the database in 60 seconds, the operation times out.

Run-time Connection Tab

If you have checked the **Use Design-Time Connection For Run-time** checkbox in the Design-time Connection tab, most options in the Run-time Connection tab will inherit the design-time configuration and cannot be changed.

Table 4 Adapter Instance Run-time Connection Tab

Field or Button	Description
Database Name	Specify the name of the Amdocs CRM database you wish to connect to. This field is enabled only if the Use Design-time Connection For Run-time checkbox is checked in the Design-time Connection tab.
User Name	Specify a valid username to connect to the Amdocs CRM system. This field is enabled only if the Use Design-time Connection For Run-time checkbox is checked in the Design-time Connection tab.
Password	Specify a valid password. This field is enabled only if the Use Design-time Connection For Run-time checkbox is checked in the Design-time Connection tab.
Database Server Name	If you selected Microsoft SQL Server in the Configuration tab, then specify the name of the server machine. If you selected Oracle in the Configuration tab, then specify the SQL*NET connection string to the Oracle instance. This field is enabled only if the Use Design-time Connection For Run-time checkbox is checked in the Design-time Connection tab.
Database Host Name	Specify the machine name on which the database is hosted.
Database Port	Specify the port number on which the database is listening. Note: If you select Microsoft SQL Server in the Configuration tab, specify the Database Port to 1433. If you select Oracle in the Configuration tab, specify the Database Port to 1521.
JDBC Driver	The name of the JDBC driver the adapter uses to connect to the Amdocs CRM system. This field is disabled by default.
JDBC URL	The URL used by the driver to connect to the Amdocs CRM system. This field is disabled by default.

Table 4 Adapter Instance Run-time Connection Tab (Cont'd)

Field or Button	Description
Maximum Number of Reconnect Attempts	Specify the maximum number of reconnection attempts to make before the runtime adapter or adapter service is stopped. A value of -1 means reconnection attempts will continue indefinitely.
Number of Reconnect Attempts Before Suspending Impacted Service or Services	Specify the number of reconnection attempts to make before suspending a runtime adapter or adapter service. The Adapter Termination Criteria field value determines whether the adapter or an adapter service is stopped.
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)	When Any Service is Suspended stops the adapter if any one service has been unable to re-establish connection after the specified number of reconnection attempts. This field is unchangeable for TIBCO ActiveMatrix Adapter for Amdocs CRM.
Test CBO Connection	The Test CBO Connection... button allows you to validate the CBO specific parameters entered by the user (Database Server Name, Database Name, User Name, Password) by creating a test connection with the Amdocs CRM system.
Test JDBC Connection	The Test JDBC Connection... button allows you to validate the JDBC specific parameters entered by the user (Database Host Name, Database Port, User Name, Password) by creating a test JDBC connection with the Amdocs CRM system.

Adapter Services Tab

This tab is visible only when you check the **Show All Tabs** checkbox in the Configuration tab.

Specify common parameters to one or more types of services.

Click **Apply** to apply the changes before leaving this tab.

Table 5 Adapter Instance Adapter Services Tab

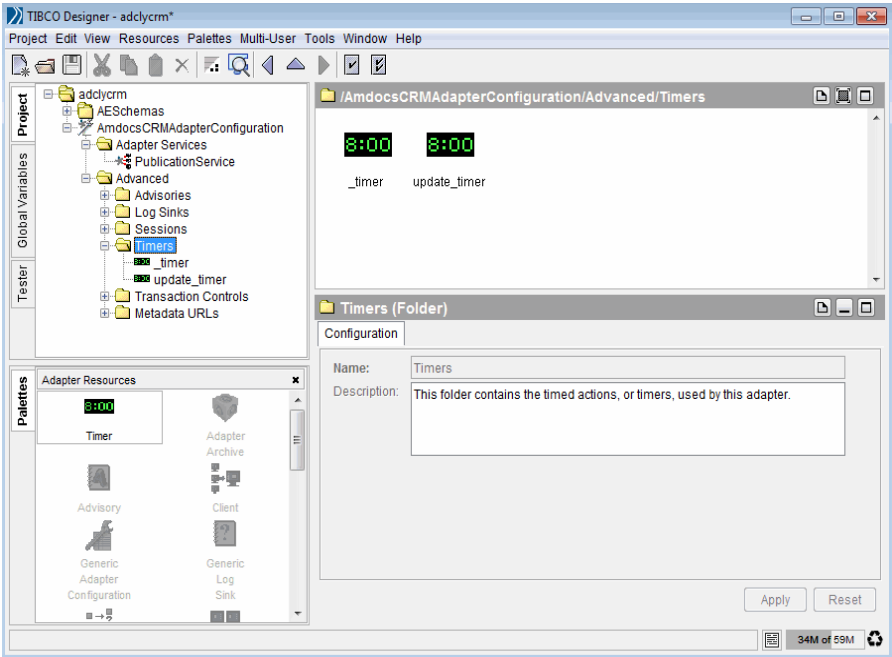
Field	Description
Retain Published Messages	<p>On checking this checkbox, the published messages will be deleted from the TIBCO Staging tables. By default, this checkbox is checked and the published messages will not be deleted from TIBCO Staging tables.</p> <p>Warning: Deletion of messages is not recommended because retracing the history of the messages is impossible. But performance of the adapter may degrade if the size of the TIBCO Staging tables grows too large.</p>
Polling Batch Size	<p>This field is used to specify the number of unprocessed messages to be processed while polling the database. The default polling size is 10.</p>

Changing the Polling Interval and the Update Interval

To change the polling interval and the update interval, complete the following steps:

1. Navigate to the Timers folder available in the Advanced folder of the project panel.
These two timer icons available: _timer icon and update_timer icon.
2. Select the **_timer** icon to specify appropriate values in the following fields:
 - Startup State—The Startup State refers to the state of the timer during startup. Select the Startup State as Active OR Inactive from the drop-down list as required.
 - Interval (milliseconds)—In this field, enter the interval after which the adapter polls the database repeatedly.
 - Repeating—Select the Repeating checkbox if this is a repeating timer. If the checkbox is cleared, this is an once-only timer.

Figure 7 Change the polling interval



- 3. Select the **update_timer** icon to specify the update interval. In the Interval (milliseconds) field, enter the interval after which the adapter updates the database repeatedly.

General Tab

This tab allows you to set a termination subject or topic and specify the encoding type. Note that the adapter should communicate only with other applications that support the same code pages or Unicode.

Click **Apply** to apply the changes before leaving this tab.

Table 6 Adapter Instance General Tab

Field	Description
Termination Subject or Topic	<p>A message sent on the termination subject (if Rendezvous is the transport) or topic (if JMS is the transport) stops the adapter. In most cases, you should use the default value.</p> <p>See <i>TIBCO Rendezvous Concepts</i> for information about specifying subject names. See <i>TIBCO Enterprise Message Service User's Guide</i> for information about publishing on a topic.</p>

Table 6 Adapter Instance General Tab (Cont'd)

Field	Description
Pre-Publication	<p>This field is used to implement pre-publication User Exits for the Publication Service. Enter the event name and class name separated by a colon and each pair should be separated by a semi colon.</p> <p>Example:</p> <pre>update_case:prepubevent;TEST.CASE:prepubevent</pre>
Pre-Subscription	<p>This field is used to implement pre-subscription User Exits for the Subscription Service. Enter the event name and class name separated by a colon and each pair should be separated by a semi colon.</p> <p>Example:</p> <pre>TEST.CASE:presubevent</pre>
Post-Subscription	<p>This field is used to implement post-subscription User Exits for the Subscription Service. Enter the event name and class name separated by a colon and each pair should be separated by a semi colon.</p> <p>Example:</p> <pre>TEST.CASE:postsubevent</pre> <p>For more details on User Exits, see User Exits on page 223.</p>
Number of Threads	<p>This field is used to implement pre-publication User Exits for the Publication Service. Enter the event name and class name separated by a colon and each pair should be separated by a semi colon.</p> <p>Note:</p> <ul style="list-style-type: none"> You need to determine the maximum value for the number of threads after doing the necessary performance tests on your hardware/software production environment. To publish the messages in an ascending order of message numbers, you must configure the thread count as 0 or 1.

Logging Tab

Use these settings to configure a log file or log sinks, including which types of trace messages you want logged and where they are sent.

Click the **Apply** button to apply the changes before leaving this tab.

Table 7 Adapter Instance Logging Tab

Field	Description
Use Advanced Logging	<p>When unchecked, the standard log file is used. This is the default. Fill out the remaining fields on this tab. You do not need to read the rest of this field description.</p> <p>When checked, you can set two standard output destinations (sinks) for trace messages and set the tracing level for the roles selected. The following sink types are available:</p> <ul style="list-style-type: none">• File• STDIO• Hawk• Network <p>See Creating Log Sinks on page 27 for more information.</p>
Log to Standard I/O	<p>When checked, trace messages are displayed in the command prompt window where the adapter is started. This is the same as creating a STDIO sink. When unchecked, trace messages do not display in the window.</p>
Log File	<p>Specify the name of the log file to which trace messages are written. This is the same as creating a file sink. Global variables can be used to specify the location of the log file. For details, see Using Global Variables on page 126.</p> <p>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.</p>
Log Info/ Debug/ Warning/ Error Messages	<p>Trace messages of the selected level or levels will be collected in the named log sink. You can configure which levels of trace messages you want logged, and to where trace messages are sent.</p> <p>There are three types of logs (log sinks) that you can configure to hold trace messages, corresponding to three levels (roles) of trace messages, Info, Warning and Error.</p> <p>A fourth level of trace messages, Debug, is reserved and should not be enabled unless requested by the TIBCO Product Support Group. The Debug option writes a lot of information to the log file and significantly degrades the performance of the adapter.</p>

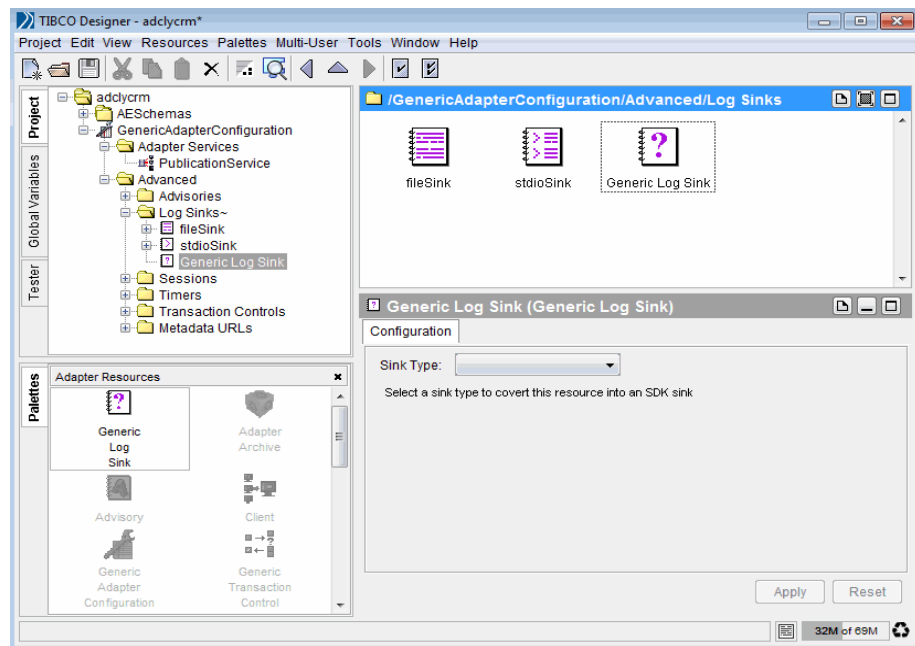
Creating Log Sinks

When you check the **Use Advanced Logging** checkbox, you configure log sinks using icons in the TIBCO Designer project panel. This gives you complete control on selecting the destinations and associating desired roles with each of the destinations.

To create the log sinks, follow these steps:

1. Check the **Use Advanced Logging** checkbox.
2. Click the **Apply** button.
3. In the TIBCO Designer project panel, select the **Log Sinks** folder under the **Advanced** folder.

Figure 8 Generate sinks



4. Select an existing log sink or create a new one:
 - Selecting the File or STDIO log sink icon.
 - Creating a new log sink by dragging and dropping the **Generic log sink** icon from the palette panel into the design panel, then assigning a type to it from the drop down menu in the configuration panel. Click the **Apply** button.

5. With the desired log sink icon selected in the design panel, fill in the fields in the configuration panel. You can also change the name and enter a description for each sink by right-clicking on the sink icon in the project panel.
- File sink logs the trace messages to a file. Specify the file limit, file count, and the option to append or overwrite. By default, the file limit is 30000 bytes, the file count is 3, and the mode is append.

— STDIO sink sends trace messages to stdout OR stderr. By default, stdout is selected.

— Hawk sink sends each trace message to TIBCO Hawk Monitor or TIBCO Hawk Display using the Hawk session, which is created by the adapter for monitoring purposes. Specify the MicroAgent Name. \

— Network sink publishes each trace message on TIBCO Rendezvous. Specify the session and the subject on which the trace messages need to be published.

Startup Tab

This tab displays the startup behavior.

Table 8 Adapter Instance Startup Tab

Field	Description
Show Startup Banner	The startup banner displays the runtime adapter version, the infrastructure version on which the adapter is built, and copyright information in the console window when the adapter is started. Check this checkbox to display the startup banner.
Metadata Search URL	This field is predefined and cannot be changed. It specifies the location where the adapter searches for base schemas. All schemas that have been defined and saved at this location are loaded at startup.

Monitoring Tab

These settings do not need to be configured unless TIBCO Hawk is installed.

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.

Click the **Apply** button to apply the changes before leaving this tab.

Table 9 Adapter Instance Monitoring Tab

Field	Description
Enable Standard Microagent	Allows you to turn on or off the standard TIBCO Hawk Microagent. Clicking the globe icon changes the checkbox to a text field, allowing you to specify a global variable. When this is a text field, turn the microagent on and off by entering true or false.
Standard Microagent Name	<p>This is the name for the standard microagent to be registered with the TIBCO Hawk system. In most cases, the default value is used.</p> <p>The InstanceId variable need not be set because it is automatically set at run time by the runtime adapter.</p>
Standard Microagent Timeout (ms)	The timeout value for the Standard MicroAgent in milliseconds.
Enable Class Microagent	Allows you to turn on or off the instance-specific or class-specific standard TIBCO Hawk Microagent. You can configure how the class microagent is turned on and off in this field: clicking the globe icon changes the checkbox to a true/false text field.
Class Microagent Name	This is the name for the class microagent to be registered with the TIBCO Hawk system. In most cases, the default value is used.
Class Microagent Timeout(ms)	The timeout value for the Class MicroAgent in milliseconds.
Default Microagent Session	<p>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.</p> <p>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.</p>

Chapter 4 **Adapter Services Options**

This chapter introduces the adapter services configured for an adapter instance. All the tasks are performed using TIBCO Designer

Topics

- [Overview of Adapter Services Options, page 32](#)
- [Publication Service, page 36](#)
- [Subscription Service, page 40](#)
- [Request-Response Service, page 44](#)
- [Request-Response Invocation Service, page 48](#)

Overview of Adapter Services Options

TIBCO ActiveMatrix Adapter for Amdocs CRM supports Publication, Subscription, Request-Response, and Request-Response Invocation services.

After configuring an adapter instance, select the adapter instance, then click **Adapter Services**. Add one or more services to the adapter instance by dragging a **Publication Service**, **Subscription Service**, **Request-Response Service** or **Request-Response Invocation Service** icon from the palette panel to the design palette.

Setting Schema

You can set schema parameters for the services by fetching the CBOs.

To fetch the CBOs in the Fetch CBO tab, follow these steps:

1. Fetch the Main CBO. You can do either of the two ways:
 - Enter the CBO name in text field and click the **Fetch CBO** button.
 - Click the **Fetch CBO List...** button and select the Main CBO in the list displayed. Click the **OK** button.
2. Click the **Fetch Contained CBO...** button to fetch the Contained CBO from the list.
3. Click the **Apply** button.

Message Transports

The transport type (Rendezvous or JMS) you select for the runtime adapter determines which quality of service, delivery mode, connection factory type, and wire format the service can use.

Quality of Service

This is the level of service that determines how messages are sent.

Possible values are:

- **Reliable**—(TIBCO Rendezvous transport type only) Reliable message delivery.
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. Messages are received without explicit confirmation.

- **Certified**—(TIBCO Rendezvous transport type only) Certified message delivery.
Offers stronger assurances of message receipt, along with tighter control, greater flexibility and fine-grained reporting. 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.

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**—(TIBCO Rendezvous transport type only) A distributed queue is a group of cooperating transport objects, each in a separate process.

To obtain load balancing among servers, the adapter uses 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*.
- **Transactional**—This option is no longer supported. The option is only included for backward compatibility.

Delivery Mode

(JMS transport type only) The method of delivery for a JMS message. The semantics for these fields are somewhat more complex than the explanation given here. See *TIBCO Enterprise Message Service User's Guide* for more information.

- For Publication and Request-Response Invocation service, the delivery modes are:
 - **Persistent**—In general, a message marked persistent will be available to a JMS client even if the TIBCO Enterprise Message Service server goes down. Persistent messages are held in secondary storage in the server and have guaranteed delivery when sent to a topic that has durable subscribers. (If a topic has no durable subscribers, there are no subscribers that need messages resent in the event of a server failure and therefore messages do not need to be saved.) Performance is improved because disk I/O is not required.
 - **Non-Persistent**—A message marked non persistent will not be available to a JMS client if the TIBCO Enterprise Message Service server goes down. These messages are never written to persistent storage.

- For Subscription and Request-Response service, the delivery modes are:
 - Durable—Indicates that the service is registered with the EMS server.
Messages sent to a durable Subscription Service are held by the EMS server until they are consumed by the service. The service can be down and expect to receive its messages when it comes back up.
 - Non-durable—Indicates that the service is not registered with the EMS server.
Messages sent to a non-durable Subscription Service are not held by the EMS server. If the service is down, it will not receive the messages that arrived at the EMS server while the service was down.

Connection Factory Type

(JMS transport type only) The JMS mode of transport supports the following messaging protocols (TIBCO Enterprise Message Service must be installed to use the JMS transport):

- Topic—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 type of message protocol is also known as broadcast messaging because messages are sent over the network and received by all interested subscribers. This messaging model is known as publish-subscribe.
- Queue—A message sent to a queue is consumed by one receiver.
Each message has only one receiver though multiple receivers may connect to the queue. The first receiver to access the queue receives the message and the other receivers do not. A queue retains all messages sent until a certain time the messages are consumed or expired. This message protocol is known as point-to-point.

Wire Formats

Services must use the same wire format to exchange data.

- Rendezvous Message (TIBCO Rendezvous transport type only)—A self-describing wire format used by TIBCO Rendezvous applications.
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.
- XML Message (TIBCO Rendezvous and JMS transport types)—The XML Message wire format conforms to specifically constructed and fully compliant XML Schema (XSD) based on the existing definition of the TIBCO ActiveEnterprise schema.
- ActiveEnterprise Message (TIBCO Rendezvous transport type only)—An externally-described XML wire format supported by the TIBCO Adapter SDK.

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

Publication Service

The Publication service sends messages from an Amdocs CRM application to the applications configured for the TIBCO environment.

You can configure the parameters for the Publication service using the following tabs:

- [Configuration Tab, page 36](#)
- [Fetch CBO Tab, page 37](#)
- [Transport Tab, page 37](#)
- [Schema Tab, page 39](#)

Configuration Tab

You can specify a name and select the transport type for a Publication service in this tab. Click the **Apply** button to apply the changes before leaving this tab.

Table 10 Publication Service: Configuration Tab

Field	Description
Name	<p>You can use the default name or replace it with a name of your choice.</p> <p>A service name must use alphanumeric characters. An underscore (<code>_</code>) character can be used. The entire instance name must be less than 80 characters. The space character cannot be used in an instance name.</p> <p>A service name cannot use global variables.</p>
Description	<p>Provide information about the Publication service. This field is optional.</p>
Transport Type	<p>Select the transport type (JMS or TIBCO Rendezvous) to be used by the runtime adapter. This selection determines which options appear in the Transport tab.</p> <p>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.</p> <p>To enable and configure SSL, in the Project panel, expand the Advanced folder, then expand the <i>Sessions</i> 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.</p>

Configuring Multiple Publication Service Endpoints

The publishing component of the adapter allows you to define multiple Publication Service endpoints in an adapter instance. This enables quicker message processing.

For the adapter to sort and publish messages from multiple endpoints, the subject names of messages specified by the ClearBasic or Web forms must be the same as the subject names configured in the project. Otherwise, the Publication Service will handle the messages using a default Publication Service endpoint that publishes the message with the subject name specified by the ClearBasic form.

Fetch CBO Tab

This tab can be used to set schema parameters for Publication service by fetching the CBOs included in a message. For more details on how to fetch the CBOs in the Fetch CBO tab, see [Setting Schema on page 32](#).



- You need not set any options in the Fetch CBO tab if you intend to use generic schema for publishing a message. See [page 55](#) for details.
- In case of a change in schema in Amdocs CRM, the schema needs to be fetched and loaded again.

Transport Tab

Message Transport options can be set for the Publication service depending on the transport type selected in the Configuration tab.

Click **Apply** to apply the changes before leaving this tab.

Table 11 Publication Service: Transport Tab (Sheet 1 of 3)

Field	Description
When TIBCO Rendezvous is selected as the transport type, the following options are available.	
Message Subject	Message subject for this Publication Service.
Reply Message Subject	Reply destination of this Publication Service.
Quality of Service	Select the level of service that determines how messages are sent. See Quality of Service for a description of these options. <ul style="list-style-type: none">• Reliable• Certified

Table 11 Publication Service: Transport Tab (Sheet 2 of 3)

Field	Description
Wire Format	<p>The wire format in which data will be sent. See Wire Formats for a description of these formats.</p> <ul style="list-style-type: none"> ActiveEnterprise Message XML Message
Session Reference	<p>When you create a service, TIBCO Designer creates a corresponding session resource in the Sessions folder of the Advanced folder and displays it in this field.</p> <p>If you have explicitly created a custom session of the same type, you can click the Browse resources... button to replace the autogenerated session. It is recommended not to change the session for a service.</p>
Endpoint Reference	<p>An endpoint reference for the service. This is a disabled field and points to the corresponding endpoint resource in the Sessions folder of the Advanced folder. The endpoint resource is automatically created by TIBCO Designer.</p>
When JMS is selected as the transport type, the following options are available.	
Destination	<p>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.</p> <p>You can override the default dynamic destination by specifying the static destination in this field. The static destination must be defined on the EMS server before it can be used by the runtime adapter.</p> <p>See <i>TIBCO Enterprise Message Service User's Guide</i> for information about destinations.</p>
Reply Destination	<p>Reply destination of this Publication Service.</p>
Wire Format	<p>The wire format in which messages are to be published. See Wire Formats for a description of the format.</p> <ul style="list-style-type: none"> XML Message
Connection Factory Type	<p>Select one of the following:</p> <ul style="list-style-type: none"> Topic Queue <p>See Connection Factory Type for a description of connection factory type.</p>

Table 11 Publication Service: Transport Tab (Sheet 3 of 3)

Field	Description
Delivery Mode	<p>The delivery mode for each message sending operation. See Delivery Mode for a description of each mode.</p> <ul style="list-style-type: none"> • Persistent • Non-Persistent
Session Reference	<p>When you create a service, TIBCO Designer creates a corresponding session resource in the Sessions folder of the Advanced folder and displays it in this field.</p> <p>If you have explicitly created a custom session of the same type, you can click the Browse resources... button to replace the autocreated session. It is recommended not to change the session for a service.</p>
Endpoint Reference	<p>An endpoint reference for the service. This is a disabled field and points to the corresponding endpoint resource in the Sessions folder of the Advanced folder. The endpoint resource is automatically created by TIBCO Designer.</p>

Schema Tab

The Schema tab can be used to view the schema for the Publication service.

Table 12 Publication Service: Schema Tab

Field	Description
Class Reference	<p>The schema class reference points to the class created for this component. Leave this field unchanged.</p>

Conditions When a Generic Schema is Used to Publish a Message

Generic schema is used to publish a message only when:

- If no Publication services are found with the subject name as that in the ClearBasicForm, the message is published using the generic schema.
- If you do not associate any CBOs with the Publication service endpoint; that is, if you do not use the Fetch CBO tab and if you directly specify the subject name in the Transport tab, the message is published using the generic schema.



TIBCO ActiveMatrix Adapter for Amdocs CRM does not support generic schema for JMS transport.

Subscription Service

The Subscription service subscribes to messages from the TIBCO environment and post them into the Amdocs CRM database.

You can configure the parameters for the Subscription service using the following tabs:

- [Configuration Tab, page 40](#)
- [Fetch CBO Tab, page 41](#)
- [Transport Tab, page 41](#)
- [Schema Tab, page 43](#)

Configuration Tab

You can specify a name and select the transport type for a Subscription service in this tab.
Click **Apply** to apply the changes before leaving this tab.

Table 13 Subscription Service: Configuration Tab

Field	Description
Name	<p>You can use the default name or replace it with a name of your choice.</p> <p>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.</p> <p>A service name cannot use global variables.</p>
Description	<p>Provide information about the Publication service. This field is optional.</p>
Transport Type	<p>Select the transport type (JMS or TIBCO Rendezvous) to be used by the runtime adapter. This selection determines which options appear in the Transport tab.</p> <p>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.</p> <p>To enable and configure SSL, in the Project panel, expand the Advanced folder, then expand the <i>Sessions</i> 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.</p>

Fetch CBO Tab

This tab can be used to set schema parameters for Subscription service by fetching the CBOs included in a message. For more details on how to fetch the CBOs in the Fetch CBO tab, see [Setting Schema on page 32](#).



- In case of a change in schema in Amdocs CRM, the schema needs to be fetched and loaded again.
- If you are using the Custom JavaBean feature of the Subscription Service, this schema can be modified to include any level of sub-classes and sequences. Any class which is defined in TIBCO Designer can be included in this schema.

Transport Tab

Message Transport options can be set for the Subscription service depending on the transport type selected in the Configuration tab.

Click **Apply** to apply the changes before leaving this tab.

Table 14 Subscription Service: Transport Tab (Sheet 1 of 3)

Field	Description
When TIBCO Rendezvous is selected as the transport type, the following options are available.	
Message Subject	Message subject for this Publication Service.
Quality of Service	<p>Select the level of service that determines how messages are sent. See Quality of Service for a description of these options.</p> <ul style="list-style-type: none"> • Reliable • Certified • Transactional • Distributed Queue
Wire Format	<p>The wire format in which data will be sent. See Wire Formats for a description of these formats.</p> <ul style="list-style-type: none"> • ActiveEnterprise Message • XML Message

Table 14 Subscription Service: Transport Tab (Sheet 2 of 3)

Field	Description
Session Reference	<p>When you create a service, TIBCO Designer creates a corresponding session resource in the Sessions folder of the Advanced folder and displays it in this field.</p> <p>If you have explicitly created a custom session of the same type, you can click the Browse resources... button to replace the autogenerated session. It is recommended not to change the session for a service.</p>
Endpoint Reference	<p>An endpoint reference for the service. This is a disabled field and points to the corresponding endpoint resource in the Sessions folder of the Advanced folder. The endpoint resource is automatically created by TIBCO Designer.</p>
When JMS is selected as the transport type, the following options are available.	
Destination	<p>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.</p> <p>You can override the default dynamic destination by specifying the static destination in this field. The static destination must be defined on the EMS server before it can be used by the runtime adapter.</p> <p>See <i>TIBCO Enterprise Message Service User's Guide</i> for information about destinations.</p>
Wire Format	<p>The wire format in which messages are to be published. See Wire Formats for a description of the format.</p> <ul style="list-style-type: none"> XML Message
Connection Factory Type	<p>Select one of the following:</p> <ul style="list-style-type: none"> Topic Queue <p>See Connection Factory Type for a description of connection factory type.</p>
Delivery Mode	<p>The delivery mode for each message sending operation. See Delivery Mode for a description of each mode.</p> <ul style="list-style-type: none"> Durable Non-Durable

Table 14 Subscription Service: Transport Tab (Sheet 3 of 3)

Field	Description
Session Reference	<p>When you create a service, TIBCO Designer creates a corresponding session resource in the Sessions folder of the Advanced folder and displays it in this field.</p> <p>If you have explicitly created a custom session of the same type, you can click the Browse resources... button to replace the autocreated session. It is recommended not to change the session for a service.</p>
Endpoint Reference	An endpoint reference for the service. This is a disabled field and points to the corresponding endpoint resource in the Sessions folder of the Advanced folder. The endpoint resource is automatically created by TIBCO Designer.

Schema Tab

The Schema tab can be used to view the schema for the Subscription service.

Table 15 Subscription Service: Schema Tab

Field	Description
Class Reference	The schema class reference points to the class created for this component. Leave this field unchanged.

Request-Response Service

The Request-Response service provides response data by executing query, workflow and customized operations in the Amdocs CRM database using CBOs.

You can configure the parameters for the Request-Response service using the following tabs:

- [Configuration Tab, page 44](#)
- [Fetch CBO Tab, page 45](#)
- [Transport Tab, page 45](#)
- [Schema Tab, page 47](#)

Configuration Tab

You can specify a name and select the transport type for a Request-Response service in this tab.

Click **Apply** to apply the changes before leaving this tab.

Table 16 Request-Response Service: Configuration Tab

Field	Description
Name	<p>You can use the default name or replace it with a name of your choice.</p> <p>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.</p> <p>A service name cannot use global variables.</p>
Description	<p>Provide information about the Publication service. This field is optional.</p>
Transport Type	<p>Select the transport type (JMS or TIBCO Rendezvous) to be used by the runtime adapter. This selection determines which options appear in the Transport tab.</p> <p>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.</p> <p>To enable and configure SSL, in the Project panel, expand the Advanced folder, then expand the <i>Sessions</i> 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.</p>

Fetch CBO Tab

This tab can be used to set schema parameters for Request-Response service by fetching the CBOs included in a message. For more details on how to fetch the CBOs in the Fetch CBO tab, see [Setting Schema on page 32](#).



In case of a change in schema in Amdocs CRM, the schema needs to be fetched and loaded again.

Transport Tab

Message Transport options can be set for the Request-Response service depending on the transport type selected in the Configuration tab.

Click **Apply** to apply the changes before leaving this tab.

Table 17 Request-Response Service: Transport Tab (Sheet 1 of 2)

Field	Description
When TIBCO Rendezvous is selected as the transport type, the following options are available.	
Message Subject	Message subject for this Publication Service.
Quality of Service	Select the level of service that determines how messages are sent. See Quality of Service for a description of these options. <ul style="list-style-type: none">ReliableCertifiedDistributed Queue
Wire Format	The wire format in which data will be sent. See Wire Formats for a description of these formats. <ul style="list-style-type: none">ActiveEnterprise Message
Session Reference	When you create a service, TIBCO Designer creates a corresponding session resource in the Sessions folder of the Advanced folder and displays it in this field. If you have explicitly created a custom session of the same type, you can click the Browse resources... button to replace the autocreated session. It is recommended not to change the session for a service.

Table 17 Request-Response Service: Transport Tab (Sheet 2 of 2)

Field	Description
Endpoint Reference	An endpoint reference for the service. This is a disabled field and points to the corresponding endpoint resource in the <i>Sessions</i> folder of the <i>Advanced</i> folder. The endpoint resource is automatically created by TIBCO Designer.
When JMS is selected as the transport type, the following options are available.	
Destination	<p>By default, a service uses a dynamic destination that is generated using the <i>Domain</i> and <i>Deployment</i> 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 <i>Domain</i> and <i>Deployment</i> are not empty.</p> <p>You can override the default dynamic destination by specifying the static destination in this field. The static destination must be defined on the EMS server before it can be used by the runtime adapter.</p> <p>See <i>TIBCO Enterprise Message Service User's Guide</i> for information about destinations.</p>
Wire Format	<p>The wire format in which messages are to be published. See Wire Formats for a description of the format.</p> <ul style="list-style-type: none"> XML Message
Connection Factory Type	<p>Select one of the following:</p> <ul style="list-style-type: none"> Topic Queue <p>See Connection Factory Type for a description of connection factory type.</p>
Delivery Mode	<p>The delivery mode for each message sending operation. See Delivery Mode for a description of each mode.</p> <ul style="list-style-type: none"> Durable Non-Durable
Session Reference	<p>When you create a service, TIBCO Designer creates a corresponding session resource in the <i>Sessions</i> folder of the <i>Advanced</i> folder and displays it in this field.</p> <p>If you have explicitly created a custom session of the same type, you can click the Browse resources... button to replace the autogenerated session. It is recommended not to change the session for a service.</p>
Endpoint Reference	An endpoint reference for the service. This is a disabled field and points to the corresponding endpoint resource in the <i>Sessions</i> folder of the <i>Advanced</i> folder. The endpoint resource is automatically created by TIBCO Designer.

Schema Tab

The Schema tab can be used to view the schema for the Request-Response service.

Table 18 Request-Response Service: Schema Tab

Field	Description
Class Reference	The schema class reference points to the class created for this component. Leave this field unchanged.

Request-Response Invocation Service

The Request-Response Invocation service requests data from external applications through the TIBCO environment.

You can configure the parameters for the Request-Response Invocation service by using the following tabs:

- [Configuration Tab, page 48](#)
- [Fetch Request CBO Tab, page 49](#)
- [Fetch Reply CBO Tab, page 49](#)
- [SubscriberOptions Tab, page 50](#)
- [Transport Tab, page 50](#)
- [Schema Tab, page 52](#)

Configuration Tab

You can specify a name and select the transport type for a Request-Response Invocation service in this tab.

Click **Apply** to apply the changes before leaving this tab.

Table 19 Request-Response Invocation Service: Configuration Tab

Field	Description
Name	<p>You can use the default name or replace it with a name of your choice.</p> <p>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.</p> <p>A service name cannot use global variables.</p>
Description	<p>Provide information about the Publication service. This field is optional.</p>

Table 19 Request-Response Invocation Service: Configuration Tab (Cont'd)

Field	Description
Transport Type	<p>Select the transport type (JMS or TIBCO Rendezvous) to be used by the runtime adapter. This selection determines which options appear in the Transport tab.</p> <p>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.</p> <p>To enable and configure SSL, in the Project panel, expand the Advanced folder, then expand the <i>Sessions</i> 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.</p>

Fetch Request CBO Tab

This tab can be used to set request schema parameters for Request-Response Invocation service by fetching the CBOs included in a message. For more details on how to fetch the CBOs in the Fetch CBO tab, see [Setting Schema on page 32](#).



In case of a change in schema in Amdocs CRM, the schema needs to be fetched and loaded again.

Fetch Reply CBO Tab

This tab can be used to set reply schema parameters for Request-Response Invocation service by fetching the CBOs included in a message. For more details on how to fetch the CBOs in the Fetch CBO tab, see [Setting Schema on page 32](#).



In case of a change in schema in Amdocs CRM, the schema needs to be fetched and loaded again.

SubscriberOptions Tab

The SubscriberOptions tab is used to define the various parameters involved in the communication between the adapter and the Amdocs CRM client, such as the Rendezvous subject name. The adapter uses these parameters to create Subscription Services dynamically at run time.

Table 20 Request-Response Invocation Service: SubscriberOptions Tab

Field	Description
Subscriber Subject	Specify the subject for the Subscription Service to be associated with this Request-Response Invocation Service.
Subscriber Session	Specify the name of the default Rendezvous session (DefaultRVSession) that is created when the adapter configuration is saved. To use another Rendezvous session, create the session first and specify the name of that Rendezvous session.
Client TimeOut (in millisec)	Specify the time interval after which the request times out. The time interval that the client has to wait for the response from the external RPC server.

Transport Tab

Message Transport options can be set for the Request-Response Invocation service depending on the transport type selected in the Configuration tab.

Click **Apply** to apply the changes before leaving this tab.

Table 21 Request-Response Invocation Service: Transport Tab (Sheet 1 of 3)

Field	Description
When TIBCO Rendezvous is selected as the transport type, the following options are available.	
Message Subject	Message subject for this Publication Service.
Quality of Service	Select the level of service that determines how messages are sent. See Quality of Service for a description of these options. <ul style="list-style-type: none">ReliableCertified
Wire Format	The wire format in which data will be sent. See Wire Formats for a description of these formats. <ul style="list-style-type: none">ActiveEnterprise Message

Table 21 Request-Response Invocation Service: Transport Tab (Sheet 2 of 3)

Field	Description
Session Reference	<p>When you create a service, TIBCO Designer creates a corresponding session resource in the Sessions folder of the Advanced folder and displays it in this field.</p> <p>If you have explicitly created a custom session of the same type, you can click the Browse resources... button to replace the autocreated session. It is recommended not to change the session for a service.</p>
Endpoint Reference	<p>An endpoint reference for the service. This is a disabled field and points to the corresponding endpoint resource in the Sessions folder of the Advanced folder. The endpoint resource is automatically created by TIBCO Designer.</p>
When JMS is selected as the transport type, the following options are available.	
Destination	<p>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.</p> <p>You can override the default dynamic destination by specifying the static destination in this field. The static destination must be defined on the EMS server before it can be used by the runtime adapter.</p> <p>See <i>TIBCO Enterprise Message Service User's Guide</i> for information about destinations.</p>
Wire Format	<p>The wire format in which messages are to be published. See Wire Formats for a description of the format.</p> <ul style="list-style-type: none"> XML Message
Connection Factory Type	<p>Select one of the following:</p> <ul style="list-style-type: none"> Topic Queue <p>See Connection Factory Type for a description of connection factory type.</p>
Delivery Mode	<p>The delivery mode for each message sending operation. See Delivery Mode for a description of each mode.</p> <ul style="list-style-type: none"> Persistent Non-Persistent

Table 21 Request-Response Invocation Service: Transport Tab (Sheet 3 of 3)

Field	Description
Session Reference	<p>When you create a service, TIBCO Designer creates a corresponding session resource in the Sessions folder of the Advanced folder and displays it in this field.</p> <p>If you have explicitly created a custom session of the same type, you can click the Browse resources... button to replace the autogenerated session. It is recommended not to change the session for a service.</p>
Endpoint Reference	<p>An endpoint reference for the service. This is a disabled field and points to the corresponding endpoint resource in the Sessions folder of the Advanced folder. The endpoint resource is automatically created by TIBCO Designer.</p>

Schema Tab

The Schema tab can be used to view the schema for the Request-Response Invocation service.

Table 22 Request-Response Invocation Service: Schema Tab

Field	Description
Class Reference	<p>The schema class reference points to the class created for this component. Leave this field unchanged.</p>

Chapter 5 **Publication Service Functionality**

This chapter describes how the Publication Service processes messages received from the Amdocs CRM system and sends the messages to applications configured for the TIBCO environment.

Topics

- [Publication Service Overview, page 54](#)
- [How the Publication Service Works, page 58](#)
- [Preparing Amdocs CRM Components for Publication Service, page 59](#)
- [Limitations of the Publication Service, page 78](#)

Publication Service Overview

The Publication Service consists of the following components:

- A set of TIBCO Staging tables (`table_tibco_message_queue` and `table_tibco_message_fields`), which stores the information to be published. The tables must be added to the Amdocs system.

TIBCO Staging tables are used to store the inserted and updated data in the Amdocs CRM system. The Publication Service periodically polls these tables and publishes the new messages. The `table_tibco_message_queue` and `table_tibco_message_fields` are the TIBCO Staging tables that contain the data for an outgoing message. For details, see [TIBCO Staging Tables on page 177](#).

- ClearBasic script that uses ClearBasic APIs to capture the data into TIBCO Staging tables. The script is executed when a user saves data to the Amdocs CRM system by clicking the **Done**, **Replace**, **Add**, or **Save** buttons on a ClearBasic form.

In case of Web client this is done through Java classes, which uses CBOs to capture data into the database.

In case of Amdocs CRM Applications, data is captured using the Data Change Event feature of Integration Gateway.

- The adapter picks these new records from the database and publishes the same to the TIBCO environment.

Publication Service Features

- The Publication Service allows the publication of events generated from both the web, LAN clients, and Amdocs CRM Applications.
- The Publication Service is multi-threaded. The number of threads can be configured.
- Multiple instances of the adapter can publish from the same Amdocs instance, providing load balance.
- Multiple Publication Service endpoints can be defined in an adapter instance.

It is recommended that the subject names (of messages to be published) specified by the ClearBasic or web forms be the same as the subject names configured in the adapter configuration. In the case of Amdocs CRM Applications, the event name configured in Data Change Event should be the same as the one specified in the adapter configuration.

- The number of records that the adapter retrieves from the Amdocs CRM system can be configured.

- The adapter can be configured to publish only a subset of attributes for the business object.
- The adapter uses the user configured CBO schema to publish the messages. If no user-defined schema is found, then the adapter uses a generic schema that contains the object name, operation type and the field name and values as name-value pairs.
- The adapter can be configured to either delete the messages in the Amdocs CRM system once they are published or change the status of the message to published.
- The adapter supports publication of schemas, which contain CBOs.

The schemas can be configured by retrieving the necessary CBOs from the Amdocs CRM system through the design-time adapter. If multiple CBO schemas of the same type have to be published in a message, they need to be given unique names. For example, to publish two address objects, they need to be named AddressR1 and AddressR2 where R1 and R2 are the current object names in the TIBCO Message Queue table.

- If one of the messages in the TIBCO Message Queue table cannot be associated with any of the configured Publication Services (if the subject names do not match), the message will be published using the default Publication Service. The subject name of the message will be the same as the subject name in the table. This message will use the generic schema.

Publication Service can be configured to publish the messages either in a generic or a CBO based schema format.

Message Formats that Use Generic Schema

Table 23 Incoming Event

Instance Name	Field Type	Field Name	Field Value Description
IncomingEvent	M_TREE	Schema	The value of schema field is a sequence of opObject instances. The structure of opObject instance in an incoming message is shown in Table 24.
	M_STRING	Name	Event name. For example, Update contact. You can give the event any name to differentiate the captured events.

Table 24 *OpObject*

Instance Name	Field Type	Field Name	Field Value Description
OpObject	M_STRING	op_type	Not used.
	M_STRING	Name	A valid Amdocs object name such as contact or site.
	M_STRING	Identifier	<p>The object instance reference name. It is used to relate this object instance to another object instance because it is possible to have multiple object instances with the same name.</p> <p>A reference name has significance only within the message. It must be unique among the message scope. It is used as a key to uniquely identify an opObject instance within a sequence of opObject instances.</p>
	M_TREE	Fields	A sequence of field instances. The structure of each field instance is shown in Table 25.
	M_TREE	ref_relations	A sequence of relation instances. The structure of each relation is shown in Table 26.

Table 25 *Field*

Field Name	Type	Value	Value Description
field	M_STRING	Name	A valid Amdocs object field name.
	M_STRING or M_INT or M_REAL	Value	M_STRING value if a Amdocs field is date-time type or string type; M_INT value if a Amdocs field is long or Boolean type; and M_REAL value if a Amdocs field is double or float type.

Table 26 Relation

Field Name	Type	Value	Value Description
relation	M_STRING	Name	A valid Amdocs relation name such as contact_role2site.
	M_STRING	Parent_ identifier	See a valid opObject instance within this message scope. The value is that of the identifier attribute of an opObject instance.

Message Formats that Contain CBOs

Table 27 Message Formats that Contain CBOs

Field Name	Type	Value	Value Description
Optype	M_STRING	n/a	Name of the operation being performed. This will be the same as the value for the operation field in the TIBCO_MESSAGE_QUEUE table.
Schema	Class	n/a	The CBO, which needs to be part of the message has to be included here.
Lookup	Sequence	n/a	Sequence of look up values.
ref_relations	Sequence	n/a	A sequence of relation instances. Each relation contains the name and parent identifier fields.

How the Publication Service Works

1. The events generated in the Amdocs CRM system are captured in the TIBCO Staging tables by ClearBasic scripts (LAN client) or JSP code (Web forms).
2. The adapter uses two ways to retrieve the new events:
 - Polls the TIBCO Staging tables periodically.
 - Uses the automatic event logging feature configured using Data Change Event editor.

The number of messages to be retrieved from the TIBCO Staging tables is configurable. The retrieved messages are then queued in memory by the adapter.

3. The threads pick up the messages sequentially from the queue and publish them using the configured Publication Service endpoints.

The Publication Service process is multi-threaded, enabling it to process messages that are picked up from the queue, quickly. You can configure the number of threads to be created by the Publication Service process in the adapter instance. However, the poller process, which retrieves the events from the TIBCO Staging tables, is single threaded.

Preparing Amdocs CRM Components for Publication Service

To prepare Amdocs CRM components for Publication Service, complete the following tasks:

1. Log in to the Amdocs CRM system.
2. [Add TIBCO Staging Tables to Amdocs CRM Database, page 59](#)
3. [Enable Publication Service for LAN Client, page 60](#)
4. [Enable the Publication Service for Amdocs CRM Web Client, page 66](#)
5. [Enable the Publication Service for Amdocs CRM Applications, page 71](#)

Add TIBCO Staging Tables to Amdocs CRM Database



- You must run the following command in the *AMDOCSCRM_HOME*\Server\dbadmin directory from the command window.
- The *tibco.sch* and *tibco.dat* files are used in the following steps. They are located in the *TIBCO_HOME*/adapter/adclycrm/*version_num*/setup directory after installation.

To add TIBCO Staging tables to Amdocs CRM database, follow these steps:

1. Get your current Amdocs CRM database schema by running the following command:

```
java -cp.; ClfySchemaMgr.jar com.clarify.schemamgr.SchemaMgr -user_name user name -password password -db_name database name -db_server server name -schema schema name
```

2. Check if you have any customized tables with type ID 502, 503, 504 or 510 in the schema file generated in [step 1](#).
 - If you do not have such tables in the schema file, directly append *tibco.sch* to the end of the generated schema file.
 - If you have such tables in the schema file, change the type ID of the tables in *tibco.sch* to other unused numbers in the range 470-511 or 2000-4999, and then append *tibco.sch* to the generated schema file.

3. Import TIBCO Staging tables to Amdocs CRM database by running the following command:

```
java -cp.;ClfySchemaMgr.jar com.clarify.schemamgr.SchemaMgr -user_name user name -password password
-db_server server name -db_name database name -upgrade upgrade schema name -fullecho -trace
Server/Info/Config/Debug -tracefile trace file name
```



This command is only used for Windows (32-bit) platforms. If you want to use this command on Windows (64-bit) platforms, you need to navigate to the *System Drive:\windows\SysWoOW64* directory and start the command window from it. Otherwise, an error occurs.

4. Import *tibco.dat* by using the *datex* utility.

```
dataex -user_name user name -password password -db_name database name -db_server server name
-imp import data
```



TIBCO Staging tables added by the Amdocs CRM system are not removed during the installation of the adapter.

Enable Publication Service for LAN Client

This section describes the procedure to enable Publication Service for the Amdocs CRM LAN client. The procedure assumes you are familiar with Amdocs User Interface Editor and ClearBasic Exchange (*cbex*). For more information, see *User Interface Editor Guide* and *A ClearBasic Programmer's Guide*.

Import ClearBasic Global Files

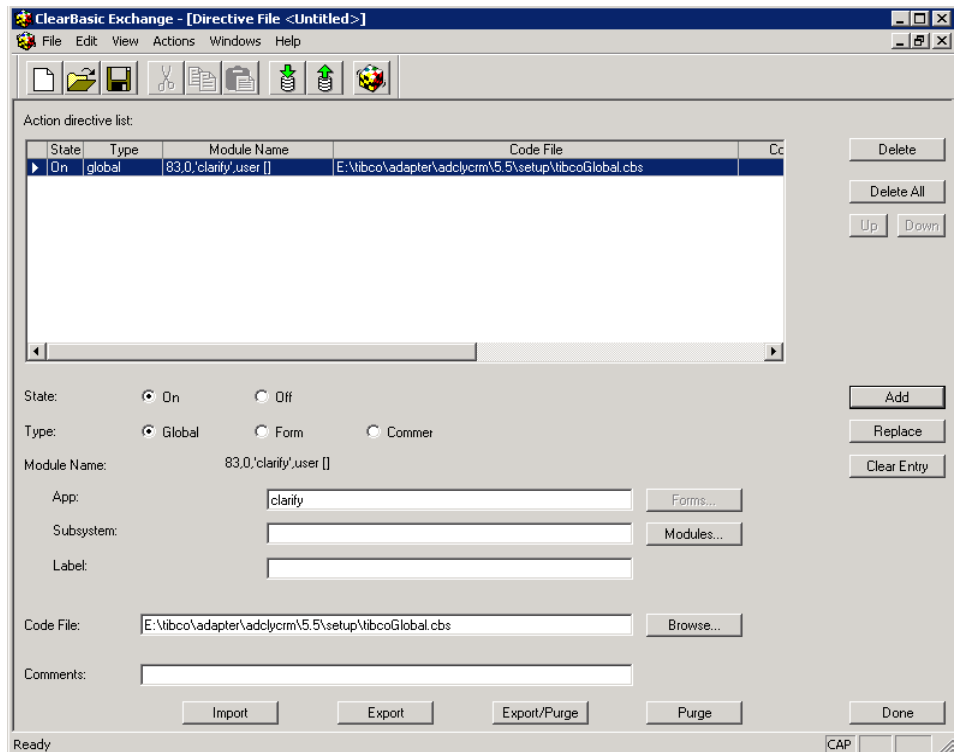
Import *tibcoGlobal.cbs* (setup directory), as follows:

1. Execute the following *cbex* command:

```
cbex -dir tibco.cbi
```

The Directive File window appears.

Figure 9 Directive File window



2. Verify that the file path is correct in the Code file column. Go to [step 3](#). If the path is not correct, use the **Browse** button to select the right path. Then click the **Replace** button. Continue with step 3.
3. Click the **Import** button.

How Publication Service Works for LAN Client

ClearBasic Global File

The adapter provides a ClearBasic Global file (`tibcoGlobal.cbs`) to facilitate the capture of business events from the Amdocs CRM system to TIBCO Staging tables. The file contains a rich set of utility functions, which can be leveraged by the user to customize the ClearBasic forms. The exposed utility functions and their descriptions are given in [Table 28](#).

Table 28 *ClearBasic Functions*

Function Name	Function Description
Sub tibcoRecordWithFilterAndSubject(rec As Record, bulks As BulkSave, ByVal currMsgNum As Long, ByVal refObjName As String, filter As List, ByVal subjectName As String, ByVal closure As String)	Inserts all fields of a Amdocs record into TIBCO Staging tables. The function will insert a row into <code>table_tibco_message_queue</code> and multiple rows into <code>table_tibco_message_fields</code> . The number of rows inserted into <code>table_tibco_message_fields</code> is the number of fields that a Amdocs object has.
Sub tibcoRecordWithSubject(rec As Record, bulks As BulkSave, ByVal currMsgNum As Long, ByVal refObjName As String, ByVal subjectName As String, ByVal closure As String)	Inserts selected fields of a Amdocs record into TIBCO Staging tables. The function will insert a row into <code>table_tibco_message_queue</code> and multiple rows into <code>table_tibco_message_fields</code> . The number of rows inserted into <code>table_tibco_message_fields</code> is the number of fields given by the filter list.
TibcoSetReferenceRelation (ByVal fromRefName As String, ByVal toRefName As String, ByVal from2toRel As String, ByVal currMsgNum As Long, bulks as BulkSave)	Sets the relation between two Amdocs records. The <code>fromRelName</code> and <code>toRelName</code> are object reference names rather than Amdocs object names. The <code>from2toRel</code> is a valid Amdocs relation name.
tibcoPublish (ByVal closure as String)	Internally, it calls <code>tibcoPublishImpl()</code> . Assign this subroutine to a button click callback subroutine. This subroutine creates its own BulkSave object.
tibcoPublish2 (Bulks As BulkSave, ByVal closure as String)	This is called by the Amdocs subroutine <code>Form_Save1()</code> . Internally, it calls <code>tibcoPublishImpl()</code> . This subroutine does not create its own BulkSave object. Instead, it uses the BulkSave object passed in by <code>Form_Save1()</code> .
tibcoPublishImpl(ByVal currMsgNum As Long, Bulks As BulkSave, ByVal closure as String)	The callback interface that must be implemented for each form that has objects to be published.

For the adapter to publish a message out of the Amdocs CRM system, write a customized ClearBasic script in a Amdocs CRM form. Each ClearBasic script is tailored for a particular form because the ClearBasic code refers to user interface control names for that particular form.

Follow these steps to write your customized ClearBasic (CB) code. It is assumed you have good ClearBasic and Amdocs UIEditor knowledge.

1. Identify the form and the operations (update or insert) that you wish to publish.
2. Save a copy of the form.
3. Write your CB code to customize the operations.
4. Save the customized CB code to the Amdocs CRM system.

In the following example, form 717 (Site form) is used to illustrate how to write your CB code to customize the site update behavior. For this example, assume you want to publish a site (all fields of site) and its related primary address, bill-to address, ship-to address (city, state, and zip code fields only) whenever the site is updated. The update operation is internally named MODIFY (see UIEditor) in this form.

Example

```
Private Sub tibcoPublishSite(ByVal closure as String)
    Dim currMsgNum As Long
    Dim BulkS As new BulkSave
    ON Error Goto ErrorHandler
    currMsgNum = tibcoNewMessageNum()
    Select Case currMsgNum
    case -1
'-- message row is not there
MsgBox "Error in fetching a new message number. Your changes will not be published.", ebOKOnly, "No Table Row"
    case Else
        call tibcoPublishImp(currMsgNum, BulkS, closure)
    End Select
    BulkS.Save
    Exit Sub

ErrorHandler:
    Dim errDesc as String
    errDesc = Err.Source & "://" & Err.Description
    MsgBox "Error:" & errDesc & "This operation will not be published", ebOKOnly, "Error"
End Sub

Private Sub tibcoPublishImp(ByVal currMsgNum As Long, BulkS As BulkSave, ByVal closure as String)
    Dim bulkR As New BulkRetrieve
    Dim billAddrList As New List
    Dim shipAddrList As New List
    Dim siteRec As New Record
    Dim primAddrRec As New Record
    Dim billAddrRec As New Record
    Dim shipAddrRec As New Record
    Dim filter As New List
```

```

Dim subjectName As String

    subjectName = "TEST.SITE" ' insert subject name here

    Set siteRec = Cobj_Location.Contents
    Set primAddrRec = Cobj_primary_add_obj.Contents

    Set siteRec = Cobj_Location.Contents
    Set primAddrRec = Cobj_primary_add_obj.Contents
    bulkr.SetRootByID "site", siteRec.GetField("objid")
    bulkr.TraverseFromRoot 0, "cust_billaddr2address"
    bulkr.TraverseFromRoot 1, "cust_shipaddr2address"
    bulkr.RetrieveRecords
    Set billAddrList = bulkr.GetRecordList(0)
    Set shipAddrList = bulkr.GetRecordList(1)
    If billAddrList.Count > 0 Then
        Set billAddrRec = billAddrList.ItemByIndex (0)
    Else
        Set billAddrRec = primAddrRec
    End If
    If shipAddrList.Count > 0 Then
        Set shipAddrRec = shipAddrList.ItemByIndex(0)
    Else
        Set shipAddrRec = primAddrRec
    End If

    filter.AppendItem "site_id"
    filter.AppendItem "name"
    filter.AppendItem "site_type"
    Call tibcoRecordWithFilterAndSubject(siteRec, BulkS, currMsgNum, "R1", filter,subjectName, closure)
    Call tibcoRecordWithSubject(primAddrRec, BulkS, currMsgNum, "R2",subjectName, closure)
    Call tibcoRecordWithSubject(billAddrRec, BulkS, currMsgNum, "R3",subjectName, closure)
    Call tibcoRecordWithSubject(shipAddrRec, BulkS, currMsgNum, "R4",subjectName, closure)
    call tibcoSetReferenceRelation("R1", "R2", "cust_primaddr2address", currMsgNum, BulkS)
    call tibcoSetReferenceRelation("R1", "R3", "cust_billaddr2address", currMsgNum, BulkS)
    call tibcoSetReferenceRelation("R1", "R4", "cust_shipaddr2address", currMsgNum, BulkS)
End Sub

Sub Form_Load
    Me.DoDefault
End sub

Sub Modify_Click
    isTriggered = false
    Me.DoDefault
    If isTriggered Then
        call tibcoPublishSite("updateSite")
    End If
End Sub

Sub Add_Click
    isTriggered = false
    Me.DoDefault
    If isTriggered Then
        call tibcoPublishSite("addSite")
    End If
End Sub

```

```
Sub Form_Save3(bs As BulkSave)
    isTriggered = true
End Sub
```

ClearBasic Script Coding Conventions

This section describes the ClearBasic script coding conventions for writing objects into TIBCO Staging tables. Follow these steps for each form with objects to be published.

1. Declare the global functions in the beginning of each form.

```
Declare Sub tibcoRecordWithSubject(rec As Record, bulks As BulkSave, ByVal currMsgNum As Long, ByVal refObjName As String, ByVal subjectName As String, ByVal closure As String)
```

```
Declare Sub tibcoRecordWithFilterAndSubject(rec As Record, bulks As BulkSave, ByVal currMsgNum As Long, ByVal refObjName As String, filter As List, ByVal subjectName As String, ByVal closure As String)
```

```
Declare Sub tibcoSetReferenceRelation(ByVal fromRefName As String, ByVal toRefName As String, ByVal from2toRel As String, ByVal currMsgNum As Long, bulks as BulkSave)
```

2. Declare a form-wide, boolean-type variable.

```
Dim isTriggered As Boolean
```

3. Find the button-click callback routines for which you need to add code, to publish a message. After identifying the callback routines, write the following lines of code:

```
isTriggered = false
Me.DoDefault
If isTriggered Then
    call tibcoPublish"yyyy"("xxxx")
End If
```

Where,

yyyy represents the object name. For example, Site and Contact are the object names.

xxxx represents the outgoing event name. For example, createContact and ModifySite are the event names.

4. Implement the tibcoPublishImp() routine. Code the logic to selectively publish a message. This routine is called by tibcoPublishyyyy() internally. In this routine, call tibcoRecord(), tibcoRecordWithFilter(), and tibcoSetReferenceRelation() to write your Amdocs records into TIBCO Staging tables.

5. Write the following line of code into Form_Save3().

```
isTriggered = true
```

Use the above code to fix a Amdocs bug. Me.DoDefault always returns 0, whether the transaction in Me.DoDefault succeeds or fails. Do not add rows into TIBCO Staging tables if a transaction is rolled back (fails).

By setting `isTriggered` to `true` in `Form_Save3()`, you enable the callback to help determine if a transaction is committed, because only committed transactions will launch this callback routine.



The above approach works in most forms. However, some Amdocs baseline forms (example, 672) use `Clfy_Form_Save1()` internally. In such cases, the above approach will not work. Instead, you must call `tibcoPublishyyyyy (BulkSave, 'xxxx')` in `Form_Save1()`, where, `yyyyy` represents the object name, and `xxxx` represents the outgoing event name.

Enable the Publication Service for Amdocs CRM Web Client

This section describes the procedure to enable the Publication Service for the Amdocs CRM Web client. This procedure assumes you are familiar with Java Server Pages and BEA Weblogic. For more information, see *Web Application Implementation Guide*.

How the Publication Service Works for Amdocs CRM Web Client

In the case of web forms, the Publication Service uses JSP code to selectively write business object values to TIBCO Staging tables, which the Publication Service polls periodically. For details, see [TIBCO Staging Tables on page 177](#).



- The adapter supports the following web applications with Oracle database:
 - Amdocs CRM eOrder
 - Amdocs CRM eSupport
- If the Weblogic you installed is using the previous version of Java 6, you need to install Java 6 before creating an order and update the path of Java 6 to the `setDomainEnv` file, which is located in the `Weblogic_domain_home\bin` directory. Where, `Weblogic_domain_home` is the directory of Weblogic domain.

Global Java Class

The Java utility methods used for selectively writing CBO values to TIBCO Staging tables are in `TibcoCboController.class`. You can place this class into the Web server classpath to access this object inside your JSP forms.

In all cases, only six utility methods are required to write rows into TIBCO Staging tables, as listed in [Table 29](#).

Table 29 Java Methods

Method Name	Method Description
String generateCurrentMessageNum (Base bobase)	Generates and returns a new unique message number to be inserted in the TIBCO Staging tables.
void tibcoSetReferenceRelation (Base bobase, String fromRefName, String toRefName, String from2toRel, String curMsgNum)	Sets the relation between two Amdocs records. fromRelName and toRelName are object reference names rather than Amdocs object names. from2toRel is a valid Amdocs relation name.
void tibcoSetTraversalRelation (Base bobase, String fromRefName, String toRefName, String to2FromRel, String curMsgNum)	Sets the traversal relation between two Amdocs records. The fromRelName and toRelName are object reference names rather than Amdocs object names. The from2toRel is a valid Amdocs relation name.
void tibcoRecordWithSubject (Base bobase, String operation, String refObjectName,String subjectName, String curMsgNum)	Inserts all fields of a Amdocs record into TIBCO Staging tables. The function will insert a row into table_tibco_message_queue and multiple rows into table_tibco_message_fields. The number of rows inserted into table_tibco_message_fields are the number of fields that a CBO has.
void tibcoRecordWithFilterAndSubject (Base bobase, String operation, String refObjectName, String[] filter, String subjectName, String curMsgNum)	Inserts selected fields of a record into TIBCO Staging tables. The function will insert a row into table_tibco_message_queue and multiple rows into table_tibco_message_fields. The number of rows inserted into table_tibco_message_fields are the number of fields given by the filter list.
void tibcoCommit()	Reflects the changes made by TIBCO methods to the TIBCO Staging tables in a single batch transaction.

For the adapter to publish a message out of the Amdocs CRM system, write a customized JSP code in a Amdocs CRM web form. Each JSP code is tailored for a particular form because the JSP code refers CBOs for that particular form.

Follow these steps to write your customized JSP code. It is assumed you have good CBO programming knowledge.

1. Identify the JSP and the operations that you wish to publish.
2. Save a copy of the form.

3. Write your JSP code to customize the operations.
4. Replace the original JSP with the customized JSP.

In the following example, an excerpt from `reg_do_action_1.jsp` is used to illustrate how to write your JSP code to customize the Case creation behavior. In this example, assume you want to publish a Case, its related address, site and contact information whenever the Case is created.

```
<%!
else if(strSubmit1.equals("/XLAT"/"Submit To CSR"))

{
    strAction="SubmitToCSR";

    MyCaseId = RegisterDoAction.doCreateCase(ClfyForm, ClfySession, session, request, cboWebSupStr,errorMsg);

    String strRedirectTo;

    if ( MyCaseId.equals(""))

        strRedirectTo="../CommonJSP/ErrorPage.jsp";

    else

    {

        //CUSTOMIZED CODE BEGINS HERE..

        try

        {

            com..cbo.Case boCase = null;

            boCase = (com..cbo.Case) ClfyForm.createBO("com..cbo.Case");

            boCase.addNew();

            boCase.setFilter("id_number="+MyCaseId+"");

            boCase.setDataFields("");

            boCase.setBulkName("bulk");

            boCase.getBulk().query();

            TibcoCboController tc= new TibcoCboController();

            String currMsgNum = tc.generateCurrentMessageNum(boCase);

            tc.tibcoRecordWithSubject(boCase,"insert","R1","TEST.CASE", currMsgNum);

            tc.tibcoRecordWithSubject(boCase.getContact(),"insert","R2","TEST.CASE", currMsgNum);

            tc.tibcoSetReferenceRelation(boCase,"R1", "R2", "case_reporter2contact", currMsgNum);
```



```

tc.tibcoRecordWithSubject(boCase.getSite(),"insert","R3","TEST.CASE", currMsgNum);

tc.tibcoSetReferenceRelation(boCase,"R1", "R3", "case_reporter2site", currMsgNum);

tc.tibcoRecordWithSubject(boCase.getAddress(),"insert","R4","TEST.CASE", currMsgNum);

tc.tibcoSetReferenceRelation(boCase,"R1", "R4", "case2address", currMsgNum);

tc.tibcoCommit();

}

catch(Exception tibException)

{

System.out.println("Tibco Error: " + tibException);

errorMsg.setDefaultStatusTitle("TIBCO_TEST.CASE_ERR");

errorMsg.setSpecialStatusMessage(tibException.toString());

}

//CUSTOMIZED CODE ENDS HERE

}

strRedirectTo="reg_thanksb2b.jsp?CaseNum="+ cboWebSupStr.URLEncode(session,MyCaseId);

response.sendRedirect(strRedirectTo);

}

```

JSP Coding Conventions

This section describes the JSP coding conventions for writing objects into TIBCO Staging tables. Follow these steps for each JSP with objects to be published.

1. Import the `TibcoCboController.class` in the beginning of each form.

```

<%@ page import="com..cbo.*, com.tibco.cbo.TibcoCboController, com..common.*"
errorPage="/CommonJSP/ErrorPage.jsp" %>

```

2. Identify the methods inside the JSPs, where the Amdocs objects are updated or inserted using `updateAll()` or `update()` methods of the business objects.

`Do_CreateCase(----)` is a method inside `reg_do_action_1.jsp`, where a new case is created and is updated in the Amdocs CRM system.

3. Identify the business object, which is being updated by the JSP code and published. In this example, it is the `boCase` business object.

4. Add the TIBCO specific code after the object is updated in the database, to publish the same object.

- Create an object of `TibcoCboController` class:

```
TibcoCboController tc= new TibcoCboController();
```

- Get the next message number by passing the identified business object in step 3 to the following TIBCO method:

```
String currMsgNum = tc.generateCurrentMessageNum(boCase);
```

5. Call the other TIBCO methods to publish the contents of the identified business object and its related business objects:

```
tc.tibcoRecordWithSubject(boCase,"insert","R1","TEST.CASE", currMsgNum);
```

```
tc.tibcoRecordWithSubject(boCase.getContact(),"insert","R2","TEST.CASE", currMsgNum);
```

```
tc.tibcoSetReferenceRelation(boCase,"R1", "R2", "case_reporter2contact", currMsgNum);
```

```
tc.tibcoRecordWithSubject(boCase.getSite(),"insert","R3","TEST.CASE", currMsgNum);
```

```
tc.tibcoSetReferenceRelation(boCase,"R1", "R3", "case_reporter2site", currMsgNum);
```

```
tc.tibcoRecordWithSubject(boCase.getAddress(),"insert","R4","TEST.CASE", currMsgNum);
```

```
tc.tibcoSetReferenceRelation(boCase,"R1", "R4", "case2address", currMsgNum);
```

where:

`boCase` is the identified business object in step 3.

`R1, R2, R3, R4` are the object reference names for the Amdocs object being passed.

`insert` is the name of the operation/event.

`TEST.CASE` is the publishing subject name.

`currMsgNum` is the current message number generated in step 4.

`boCase.getContact()`, `boCase.getSite()`, `boCase.getAddress()` will return the related business objects associated with the `boCase` business object.

`case_reporter2contact`, `case_reporter2site` and `case2address` are the Amdocs object relation names.

6. Write the following line of code to save the records in a single transaction:

```
tc.tibcoCommit();
```

7. Handle the exceptions using `try` or `catch` block and provide your own implementation to handle the errors. In this case we are adding the error description into the `ErrorMsg` Object of the `reg_do_action_1.jsp`:

```
errorMsg.setDefaultStatusTitle("TIBCO_TEST.CASE_ERR");
```

```
errorMsg.setSpecialStatusMessage(tibException.toString());
```

Enable the Publication Service for Amdocs CRM Applications

The procedure described in this section assumes you are familiar with the Amdocs Data Change Event Editor tool. For more information on Data Change Event Editor tool, see the Amdocs Customization Center guide.

How the Publication Service Works for Amdocs CRM Applications

Data Change Event Editor monitors specific events occurring in the Amdocs CRM system and publishes them to an external system. Any change made to an object using Amdocs CRM Applications, generates an event, which is populated in TIBCO Staging tables. The Publication Service polls the Staging tables periodically and updates recent changes in the TIBCO Publishing table.

In order to use Data Change Event Editor, you need to access some of the JAR files packaged with the Amdocs CRM Applications. So you first need to install the Amdocs CRM Applications.

You can select the object on which the Publication service should be enabled using the Data Change Event Editor.

Defining Events and Publishers on Specific Objects

Data Change Event Editor is used to define events and publishers on specific objects. To do this, perform the following tasks:

- [Task A, Define an Event on Focus Object](#)
- [Task B, Associate a Trigger with the Event](#)
- [Task C, Define a Publisher](#)
- [Task D, Associate the Publisher with the Event](#)
- [Task E, Configure the Event for the Publisher](#)

Task A Define an Event on Focus Object

1. Start Data Change Event Editor using one of the following methods:

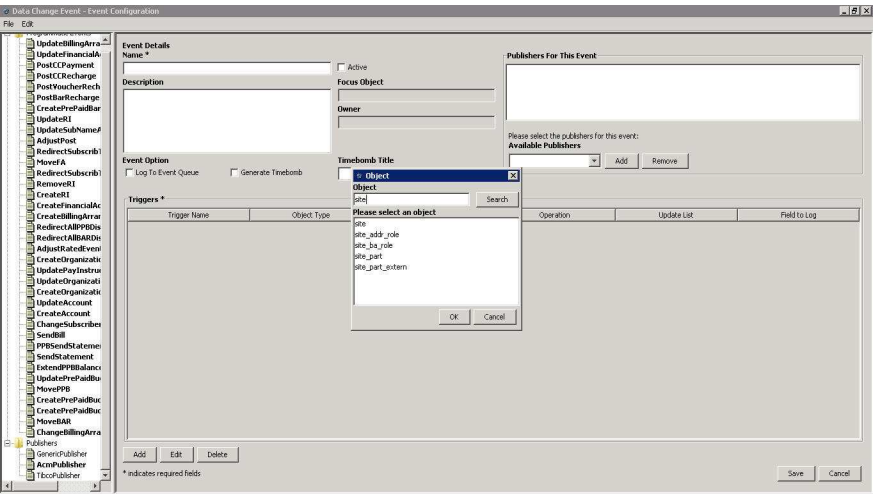
— Type the following command in the command window:

```
java [-classpath classpath] com.clarify.daevtadmin.DataChangeEventAdmin -user user -password password -dbname dbname -dbserver dbserver
```

— Run the **Start DCE.bat** file which the previous command is written in. The Start DCE.bat file is packaged with the adapter in the `TIBCO_HOME\adapter\adclycrm\version_num\setup` directory.

- 2. Define an event in the Data Change Event Editor window.
 - a. Select **Edit > New Event** from the menu to define an event based on a focus object. The Event Configuration pane and the Object dialog appear on the right panel, as shown in [Figure 10](#).

Figure 10 Event Configuration



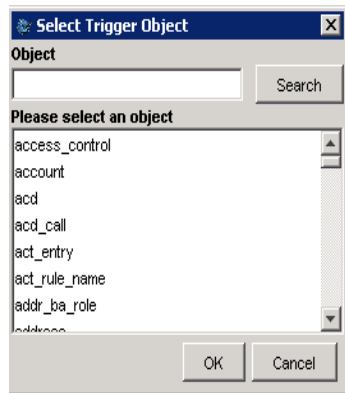
- b. Enter or select an object as the focus object, which is displayed in the Focus Object field.
 - c. Enter a name for the event in the Name field, and click the **Save** button.

The object associated with the change is referred to as the specified object. Other objects related to the focus object are known as trigger objects. An event definition defines a relationship between the focus and trigger objects. For example, you can define an event where case is the focus object and subcase is the trigger object. Any change in subcase will trigger an event to be logged in the case object. For more information, see *Integration Gateway Implementation Guide*.

Task B Associate a Trigger with the Event

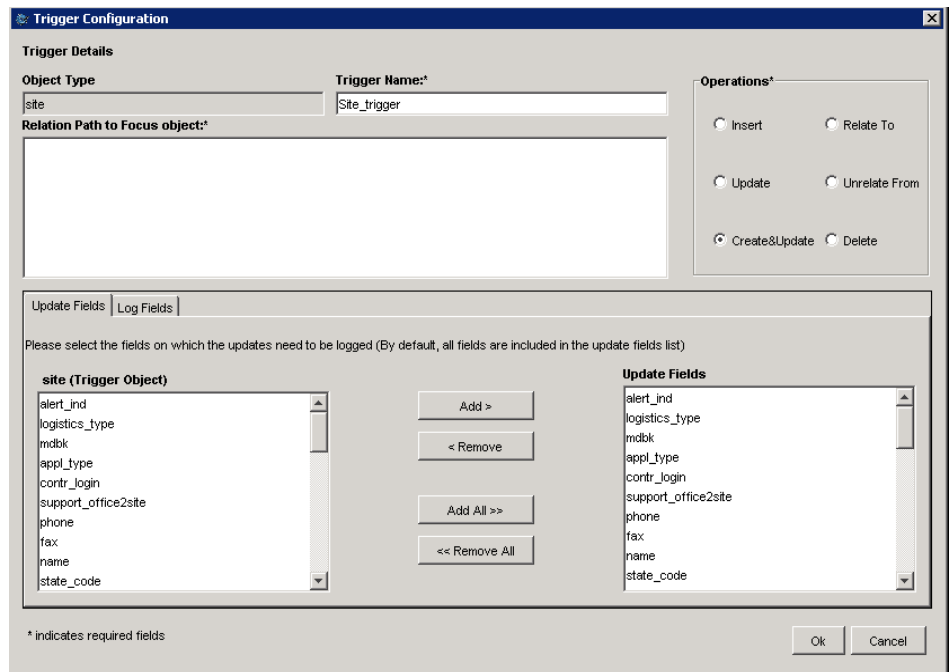
- 1. In the Event Configuration window, under the Triggers group, click the **Add** button. The Select Trigger Object dialog appears.

Figure 11 Select Trigger Object



2. Enter an object in the Object field, or click the **Search** button to choose the object you want in the Select Trigger Object dialog. Then click the **OK** button. The Trigger Configuration dialog appears.

Figure 12 Trigger Configuration



3. Enter the name of the trigger object in the Trigger Name field.

4. Specify the relation between the trigger object and focus object in the Relation Path to Focus Object field.

In case the trigger object is the same as the focus object, the relation path can be left empty. Otherwise, you must specify the relationship between the two objects. For example, to define a relationship between `acd` and `case`, you can use `acd2case` in the relation path. For more information on defining the relation path, see *Integration Gateway Implementation Guide*.

5. Select the operation on which the trigger should generate an event in the Operations field.
6. Select the items that need not be logged from the Update Fields list and click the **Remove** button.

All the items in the Trigger Object list are displayed in the Update Fields list.

7. Click the **Log Fields** tab. You can select items to be logged from both focus and trigger object.
8. Select **Focus Object** in the Available Object drop-down list. You can also select trigger or other objects based on the items you want to log into the Staging table.
9. Select the items that should be logged into the Staging table and click the **Add** button. The items displays in the Fields to Log list.
10. Click the **Ok** button to close the Trigger Configuration window. In the Event Configuration window, the trigger information appears in the Triggers group

Figure 13 Add the fields to be logged

11. Select **Active**. Active events appear in boldface in the left pane.
12. Save the changes.

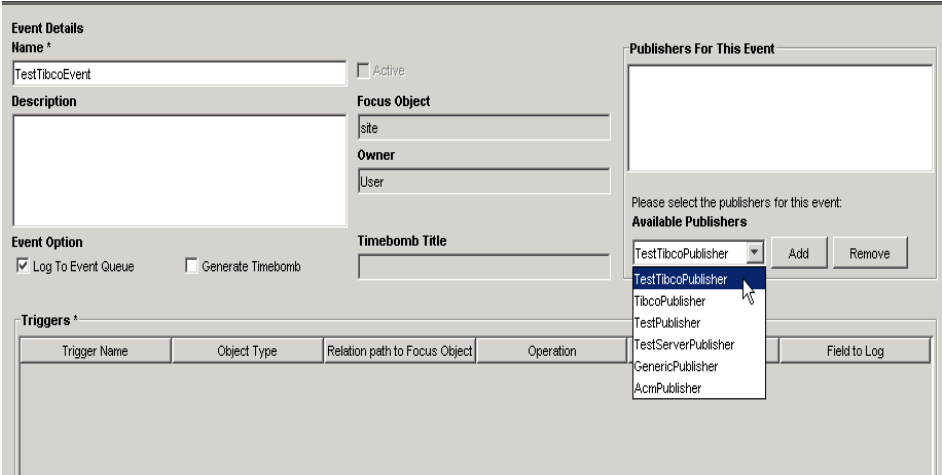
Task C Define a Publisher

1. From the left pane, right-click **Publisher** and choose **Add** from the shortcut menu. The Publisher Configuration window appears on the right pane.
2. Define a name for Publisher in the Publisher Name field. Enter `com.tibco.cbo.CustomResource` in the Initialization Class field.
3. Save the changes.

Task D Associate the Publisher with the Event

1. From the left pane, click the event you just created.
2. From the Event Configuration window, select the **Available Publishers** list. The Publisher you defined will be visible in the list.
3. Select the Publisher you defined in [Task C](#) and click the **Add** button. You will notice that the Publisher is now included in the Publishers for this Event group.

Figure 14 Associate the Publisher with the event



Task E Configure the Event for the Publisher

1. Select the Publisher associated with the event.
2. In the Publisher Configuration window, verify whether the event you defined earlier is displayed in the Publisher Events group.
3. Select the event and click the **Edit** button. The Publisher Events Configuration dialog appears.
4. In the Custom Handler Class field, enter `com.tibco.cbo.CustomPublisher`.
5. Select **Active**.
6. Click the **Ok** button to save changes.

Setting Up Event Processor

Event Processor is a command-line Java Program that is used to process and publish logged events to resources.

To set up Event Processor, follow these steps:

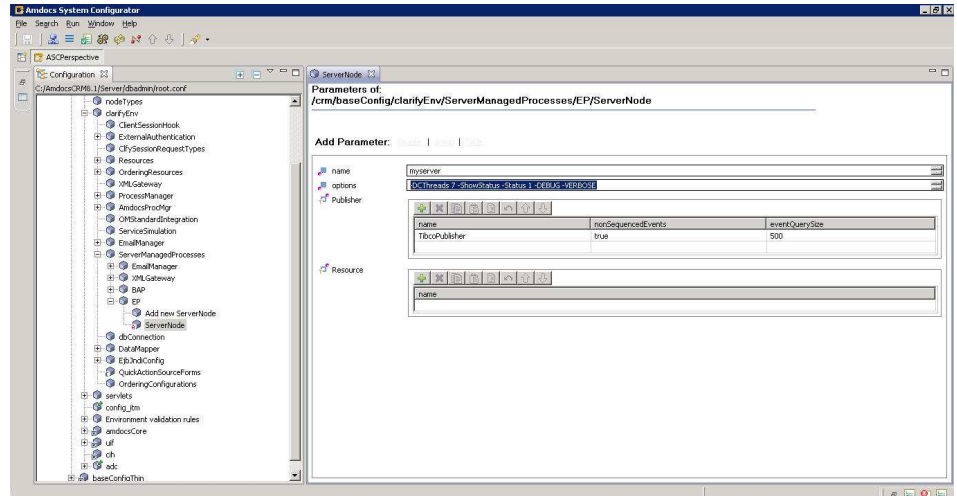
- [Task A, Specify the Instances of Event Processor](#)
- [Task B, Modify the Running File](#)
- [Task C, Run Event Processor](#)

Task A Specify the Instances of Event Processor

1. Start Amdocs System Configurator.

- Expand **crm > baseConfig > Env > ServerManagedProcesses > EP** in the Configuration tree in the left pane. Click the **ServerNode** object, and the ServerNode pane appears on the right, as shown in [Figure 15](#).

Figure 15 Amdocs System Configurator



- Enter a name in the Name field.
- Specify the parameters in the Options field. See *Amdocs CRM Integration Gateway Implementation Guide* for more information.
- Add the publisher name defined in [Define a Publisher on page 75](#) in the Publisher field.

Task B Modify the Running File

After specifying the instances of Event Processor, you need to complete the following to run Event Processor:

- Unzip the ClfyEvtProc.ear file from the *AMDOCSCRM_HOME* directory to the *AMDOCSCRM_HOME\ClfyEvtProc* directory.
- Modify the values of *JDK_DIR*, *J2EE_JAR* and *JARDIR* specified in the *Run_event_processor.bat* file according to the runtime environment.

The *Run_event_processor.bat* file is in the *TIBCO_HOME\adapter\adclycrm\version_num\setup* directory.

Task C Run Event Processor

Run Event Processor from the command window using the modified *Run_event_processor.bat* file.

Limitations of the Publication Service

The following are the limitations of the Publication service:

- The `obj_ref_name` field in TIBCO Staging table cannot contain a period (.) because the message transports (TIBCO Rendezvous and JMS) have reserved this character.
- The Publication service does not publish messages in the format received by the Subscription Service. Some minor changes like adding the wrapper class name need to be made to the message published by the Publication service for it to be received by the Subscription service.
- If the number of threads configured is more than 1, the messages may not be published in a sequence.

Chapter 6

Subscription Service Functionality

This chapter describes how the Subscription Service receives messages from the TIBCO environment and updates the Amdocs CRM system.

Topics

- [Subscription Service Overview, page 80](#)
- [How the Subscription Service Works, page 82](#)
- [Service Error Handling, page 87](#)

Subscription Service Overview

The Subscription service uses CBOs and wrappers to update or create business objects in the Amdocs CRM system. Wrappers are provided with the adapter installation for most of the CBOs. For the entire list of the wrappers provided, see Table 32 on page 82.

Subscription Service Features

The Subscription service provides the following features:

- Supports Insert and Update operations on Existing, Customized and Extended CBOs.
- Supports Delayed Acknowledgement for the RVCN quality of service.
- The Subscription service is multi-threaded. The number of threads can be configured.
- If the Subscription service fails to post a message to the Amdocs CRM system, the adapter publishes this message with a distinct error subject name.
- The Subscription service can be used to invoke a Custom JavaBean method.

The Subscription service can process any hierarchical message, including sequences and sub-classes. The incoming hierarchical message to the Subscription service can be fully accessed by the method defined in the JavaBean as a `java.util.Hashtable` object. The original structure of the incoming data is retained in the `java.util.Hashtable` object.

Subscription Service Functionality

Subscription service allows for the insert or update of Amdocs objects. The Subscription service on receiving the message from the TIBCO environment parses it and updates the Amdocs CRM system using CBOs.

CBOs are ActiveX controls and Java Bean objects, which provide an API to perform create and update operations on the Amdocs CRM system. The user has to write wrapper classes to perform these operations on business objects. For more information on Wrappers, see [Wrapper Classes on page 229](#).

All incoming messages must be an instance of `incomingEvent`. The `incomingEvent` class should have the following attributes:

Table 30 Attributes of the `incomingEvent` Class

Field Name	Type	Description
<code>optype</code>	string	Name of the operation, either <code>create</code> OR <code>update</code> .
<code>wrapper_class</code>	string	Name of the wrapper class.

Table 30 Attributes of the incomingEvent Class

Field Name	Type	Description
lookup	sequence[condition]	Used for update operation only. Helps in lookup of CBOs.
schema	class	The CBO schema object which is to be created or updated.

The Custom JavaBean functionality of the Subscription service also uses the above message structure. The relevant fields for the Custom JavaBean functionality are given next.

Table 31 Relevant Fields for the Custom JavaBean Functionality

Field Name	Type	Description
optype	string	Name of the JavaBean or Java class method to be invoked.
wrapper_class	string	<p>Name of the Java Class or JavaBean. The custom JavaBean can be given any name. But the name stored in the <code>customwrapper</code> field in the message sent to the adapter should be the name of the JavaBean prefixed with <code>TIBADCLYCUST_</code>.</p> <p>For example, if <code>LoadCaseBean</code> is the name of the java bean to be invoked, the <code>customwrapper</code> field in the message should be <code>TIBADCLYCUST_LoadCaseBean</code>.</p>
schema	class	Any class definition. For example, the class defined can include subclasses (with any level of hierarchy) and sequences. The schema need not be fetched from the database. It can be a custom configuration.

How the Subscription Service Works

On receiving a message from the TIBCO environment, the Subscription service executes a set of steps as per the following rules.

CBO Wrapper Invocation

If the `wrapper_class` field in the incoming data does not start with `TIBADCLYCUST_`, it is a CBO Wrapper invocation and the adapter executes the following steps:

1. Checks whether all the mandatory fields are present, like operation name, wrapper class name and the actual Business Object schema.
2. Uses the wrapper class specified in the incoming message to perform the operation. In case of any error, the Subscription Service uses the error Publisher to send the same incoming message with an added error attribute. See [Custom JavaBean Invocation on page 83](#) for details.
3. If there are no errors and the incoming message contains a reply subject, it sends the same message back with an error attribute set to 0.
4. Commits all the changes to the database in one transaction.

[Table 32](#) contains the CBOs for which wrapper classes have been provided with the adapter installation. For details, see [Wrapper Classes on page 229](#).

Table 32 CBOs Provided with Wrapper Classes

Main CBO	Contained CBO
ActEntry	Contact
Address	Not Required
BusOrg	Create Business Object is Not Supported
Case	Contact
Communication	edr_com_role
Contact	Site
CreditCard	Not Required
DemandDetail	Create Business Object is Not Supported
DemandHeader	Contact

Table 32 CBOs Provided with Wrapper Classes (Cont'd)

Main CBO	Contained CBO
Dialogue	Create Business Object is Not Supported
DocInst	Doc_Path
Generic	Not Required
NotesLog	Not Required
OEQuote	Not Required
Part	Part_num
PartPrice	Not Required
ServicePart	Contact
ShopList	Not Required
Site	Address
SubCase	Case
User	Employee, Site, PrivClassCache

Custom JavaBean Invocation

If the `wrapper_class` field in the incoming data starts with `TIBADCLYCUST_`, it is considered a custom JavaBean invocation.



While using the custom JavaBean invocation with Subscription service, the adapter returns a `HashTable`.

The incoming hierarchical message is passed on to the invoked method as a `java.util.Hashtable` object. The hashtable maps keys to values as defined in the incoming message to the adapter. You can iterate through the hashtable to access the various fields of the incoming data. The invoked method can be either boolean or Base type. If the invoked method is boolean type, it returns `true` for success and `false` for failure. If the invoked method is Base type, hashtable of the values are set in custom code.

In case of any error, the Subscription service uses the error Publisher to send the same incoming message with an added error attribute. If there are no errors and the incoming message contains a reply subject, it sends the same message back with an error attribute set to 0.

Ensure that the JavaBean method or the Java class method to be invoked by the adapter has `java.util.Hashtable` and `AppServer` objects as its parameters. `AppServer` class contains all the common utility functions used in the adapter and is available with the package `com.tibco.adapter.Clarity`.

Through the `AppServer` object you can access and use the CBO Session created by the adapter. The message sent to the adapter is processed and sent to the invoked method in the form of a hashtable. This hashtable retains the hierarchy structure of the incoming message.



- To access the data fields of type UI8 in the incoming data hashtable object, you need convert the fields of type UI8 to string by using the `toString()` method.

For example:

```
String fieldValue = null;

if (htIncomingData.get("fieldName") != null) {

    fieldValue = htIncomingData.get(fieldName).toString();

}
```

- Do not cast the fields of type UI8 to a String directly as follows:

```
strFieldValue=(String)htIncomingData.get("fieldName");
```

Example of Custom JavaBean Invocation

The following code is an example of a simple custom JavaBean class which is invoked through the Subscriber.

```
import java.util.*;
import com.tibco.adapter.clarity.ClarityAppServer;
/*****
* <BOCH>
* Description : This is a Custom Java Class example
* Derivation :
* Virtual members which are redefined: none
* References : ClarityAppServer. This class is available with * the adclCRM.jar that comes with the adapter
installation.
* <EOCH>
*****/

public class InvokeCase {

/*****
* <BOPH>
* Function : invoke(Hashtable htIncomingData, ClarityAppServer mClAppServer)
* Abstract : method which will be invoked by the adapter
* Parameters : ClarityAppServer - reference to ClarityAppServer Object
* Hashtable - Incoming message sent to the adapter can be accessed
* through this object
* Return Value : none
*****/
```



```

* Known Bugs and/or
* side effects : none
* Other      : none
* <EOPH>
*****/
public boolean invoke(Hashtable htIncomingData, ClarifyAppServer mClAppServer){

    //Getting a handle to FormContext object created by the adapter
    System.out.println("Formcontext object created by the adapter :"+
    mClAppServer.getFormContext());
    System.out.println("Iterating through the various keys in the      +Hashtable");
    Enumeration enum1 = htParsedClasses.keys();
    while(enum1.hasMoreElements()){
        System.out.println(enum1.nextElement());
    }
    return true;
} //end of method invoke
} //end of class

```

Applying the Custom JavaBean

To make the custom JavaBean available at runtime, follow these steps:

1. Compile the custom JavaBean to generate a CLASS file by running the following command:

On Microsoft Windows,

```

>javac -classpath
TIBCO_HOME\adapter\sdk\version_num\lib\Maverick5.jar;TIBCO_HOME\adapter\adclycrm\version_
num\lib\adclyCRM.jar;AMDOCSCRM_HOME\Server\dbadmin\ClfyCore.jar -sourcepath javafile_path -d
classfile_path javafile_name.java

```

On UNIX,

```

$javac -classpath
TIBCO_HOME/adapter/sdk/version_num/lib/Maverick5.jar:TIBCO_HOME/adapter/adclycrm/version_
num/lib/adclyCRM.jar:AMDOCSCRM_HOME/Server/dbadmin/ClfyCore.jar -sourcepath javafile_path -d
classfile_path javafile_name.java

```

2. Append the location of the generated CLASS file to the `tibco.env.CUSTOM_CP_EXT` variable in the `adclyCRM.tra` file, which is located in the `TIBCO_HOME\adapter\adclycrm\version_num\bin` directory.
3. Create a process using the Publish to Adapter activity and enter the name of the generated CLASS file in the `wrapper_class` variable in the Input tab.

The following message is a sample sent to the adapter to invoke the custom JavaBean:

```

-<MessageDataRoot>
  - otype      = invoke
  - wrapper_class = "TIBADCLYCUST_InvokeCase"

```

- schema = <this can point to any class definition>

For detailed information about how to create a process and add an activity to the process, see *TIBCO Designer User's Guide*.

Service Error Handling

The adapter supports Subscription Service error handling to handle errors that may occur during subscription. There are two options available:

- **Reply subject**—If the Subscription Service encounters an error while processing an incoming message, the message is republished along with the error using the reply subject in the subscribed message.
- **Error Publisher**—If the Subscription Service encounters an error while processing an incoming message that does not contain a reply subject, the message is republished along with the error using the Error handler Publication service endpoint configured in the project. This error publisher supports both TIBCO Rendezvous and JMS transports.

Chapter 7 **Request-Response Service Functionality**

This chapter describes how the Request-Response Service performs QUERY, WORKFLOW, and CUSTOMIZED operations using CBOs.

Topics

- [Request-Response Service Overview, page 90](#)
- [How the Request-Response Service Works, page 91](#)

Request-Response Service Overview

The Request-Response Service exposes Amdocs CRM specific operations (QUERY, WORKFLOW, and CUSTOMIZED) to the TIBCO environment thereby providing flexibility in customizing the Amdocs CRM system.

The service uses CBOs and wrappers to perform these operations. Wrappers are provided with the adapter installation for most of the CBOs. For the entire list of the wrappers provided, see Table 32. For more information on Wrappers, see [Wrapper Classes on page 229](#).

Request-Response Service Features

The Request-Response Service provides the following features:

- Supports INSERT, UPDATE, and QUERY operations on Existing, Customized, and Extended CBOs.
- Supports Delayed Acknowledgement for the RVCM quality of service.
- The Request-Response Service is multi-threaded. The number of threads can be configured.
- If the Request-Response Service fails to post a message to the Amdocs CRM system, the adapter publishes this message with a distinct error subject name.
- The Request-Response Service can be used to invoke a Custom JavaBean method.

The Request-Response Service can process any hierarchical message, including sequences and sub-classes. The incoming hierarchical message to the Request-Response Service can be fully accessed by the method defined in the JavaBean as a `java.util.Hashtable` object. The original structure of the incoming data is retained in the `java.util.Hashtable` object.

Supported Request-Response Operations

The adapter supports three types of remote Request-Response operations:

- Generic QUERY operations for querying Amdocs objects
- WORKFLOW operations to invoke workflow methods and objects
- CUSTOMIZED operations to be defined by the user

How the Request-Response Service Works

All requests received by the Request-Response Service must conform to three operational metadata classes:

- QUERY
- WORKFLOW
- CUSTOMIZED

The Request-Response Service uses incoming operation names to determine which operation it should use to serve the request. For example, if the operation name is `query`, the Request-Response Service calls a generic `QUERY` operation. All the operations use CBOs to work with the Amdocs CRM system.

All three types of Request-Response operations raise one string type of exception `RPC_SERVER_EXCEPTION`. The various error messages are included in the exception and returned to the client.

The following sections explain how the Request-Response Service works with each of the operations:

- [Query Operations, page 91](#)
- [Workflow Operations, page 93](#)
- [Customized Operations, page 96](#)

Query Operations

When the Request-Response Service receives an incoming request with the operation name as `QUERY`, it calls a generic query operation to fetch information related to the CBOs and service the incoming request.

The request object contains all the information needed to satisfy the request. The incoming request contains the name of the wrapper class (for a business object), method to be invoked, and the lookup information to identify the CBO. Using this information as input, Request-Response Service performs the required operations.

The reply object is constructed from the output of the `QUERY` operation and sent back to the calling function. It contains the input CBO along with the requested fields.

Example of Query Operation

The `QUERY` operation comes with a request to find contact information associated with a Case business object. The request object will contain lookup values for the Case business object.

The wrapper class provided for each CBO contains methods that can be used to perform a QUERY operation on that CBO.

The wrapper class invokes the `getCBO()` method of the adapter to get a handle for the Case business object to be queried. It then invokes the specific method `getContact()` to get the contact information associated with the Case.

Given next is a sample schema stored in the adapter configuration for the QUERY operation.

request_RpcQuery_query

- Input parameters

MethodName—Name of the method (`getContact`)

WrapperClass—Name of the wrapper class provided by the adapter with the package name (`com.tibco.adapter.clarify.wrapper.Case_Wrapper`)

query_object—(object of type any) Schema for the object to be queried. (Make this parameter point to the schema being queried, in this example, it is the Case object).

Class '`com.tibco.adapter.clarify.wrapper.Case_Wrapper`'.

- Output Parameters

schema of the Case Object.

'`com.tibco.adapter.clarify.wrapper.Case_Wrapper`'.

The input parameter contains the key fields, which are used to identify a Case business object and the operation name to be invoked. The reply object contains all the information that was a part of the incoming request, in addition to the contact information.

Table 33 Incoming Message Formats for Query Operations

Field Name	Type	Default	Field Value Description
MethodName	M_STRING	n/a	Name of the method to be invoked (in the wrapper class of a business object) to perform a query operation. Example: getsite, getcontact
wrapperClass	M_STRING	n/a	Name of the wrapper class. Example: com.tibco.adapter..wrapper.Case_Wrapper
query_object	any	n/a	Schema of the business object on which the query operation has to be performed. Example: /schema/case
Lookup	Sequence	n/a	Sequence of look up values.

Table 34 Outgoing Message Formats for Query Operations

Field Name	Type	Default	Field Value Description
returnValue	any	n/a	Contains the <code>query_object</code> along with the result of the query operation.
RPC_SERVER_EXCEPTION	string	n/a	If any exception or exceptions occur while performing a query operation, the server returns the exception to the calling application.

Workflow Operations

Workflow objects can also be called queue-able objects in a Amdocs CRM System. Workflow objects are the most widely used ones and they involve the most complicated business logic when they are created or changed. The usage of CBOs makes the task easier to handle.

When the Request-Response Service receives a request with its operation name as `workflow`, it calls a workflow operation to service the incoming request.

It parses the incoming message to get the request object. The request object contains all the information needed to satisfy the request. The incoming request contains name of the wrapper class (for a business object), method to be invoked, and the lookup information to identify the CBO. In addition, a workflow operation needs information about the `User Object`, `Queue Object` and `WIPBIN object`. This information forms a part of the incoming request.

The server extracts the workflow operation name from the request and invokes the operation's corresponding function, defined in the wrapper class, to meet the requirements.

The adapter uses CBOs to perform the workflow operation. The result of the operation is then returned to the calling adapter instance using a reply object. The reply object in the case of a workflow operation contains a boolean value indicating the status of the operation (True/False). In case of any error, the Request-Response Service returns an exception to indicate that the operation could not be completed.

The list of workflow operations supported is given next:

- Accept
- Assign
- Dispatch
- Forward
- Move
- Reject
- Yank

The objects on which workflow operations can be performed are listed below:

- Case
- Subcase
- OEQuote
- DemandDetail
- Dialogue

Example of Workflow Operation

Given next is a sample schema stored in the adapter configuration for a workflow operation.

request_RpcWorkflow_workflow

- Input Parameters
 - MethodName—Name of the workflow method to be invoked
(Assign, Accept, Dispatch ...)
 - wrapperClass—Name of the wrapper class
(com.tibco.adapter.clarify.wrapper.Case_Wrapper)
 - workflow_object—(Object of type any) Map this to the schema of the Case Business Object
 - lookup—This contains information about User, Queue and WIPBIN Object
 - User_LoginName—Login Name of the user
 - Queue_Title—Title of the Queue (of the logged in user)
 - isTemporary—boolean value used for Accept workflow operation
- Output Parameter
 - boolean—Status of the operation

Table 35 Incoming Message Formats for Workflow Operations

Field Name	Type	Default	Field Value Description
MethodName	M_STRING	n/a	Name of the method to be invoked (in the wrapper class of a business object) to perform a workflow operation. For example: assign, yank, accept, reject.
wrapperClass	M_STRING	n/a	Name of the wrapper class. For example: com.tibco.adapter..wrapper.Case_Wrapper

Table 35 Incoming Message Formats for Workflow Operations (Cont'd)

Field Name	Type	Default	Field Value Description
workflow_object	any	n/a	Schema of the business object on which the workflow operation has to be performed. For example: /schema/case
lookup	\schema\ workflow_lookup	n/a	Holds information about User, Queue and WIPBIN business objects. These objects are required to perform workflow operation.
User_LoginName	M_STRING	n/a	Login name of the user. Only the user who owns the object can perform workflow operation(s) on a business object. If the logged in user is not the owner, then workflow operations cannot be performed on that object. In this scenario, the adapter changes the ownership of the object to the logged in user and performs the operation.
Queue_Title	M_STRING	n/a	Title of the queue that belongs to the logged in user. This is required to perform workflow operations like forward, reject, and accept.
isTemporary	boolean	n/a	Boolean value. This is required to perform Accept workflow operation. If it is set to True, the workflow object will be temporarily accepted into the WIPBIN. After the workflow item is saved, the item automatically returns to the original queue.

Table 36 Outgoing Message Formats for Workflow Operations

Field Name	Type	Default	Field Value Description
returnValue	boolean	n/a	True OR False. Gives the status of the workflow operation performed on a workflow object.
RPC_SERVER_EXCEPTION	string	n/a	If any exception or exceptions occur while performing a workflow operation, the server returns the exception to the calling application.

Limitations of Workflow Operation

In case of a workflow operation, the input parameter contains two fields, `Login_Name` and `Queue_Title`. It should be populated with the login name of the user (one which the adapter uses to log in to the Amdocs CRM system) and the title of the queue to which the user belongs. The reason is explained below:

Certain workflow operations (reject, forward, accept) can be performed on a business object only by its owner. If the logged in user is different from the owner, then carrying out workflow operations on that object is not possible. To overcome this, the adapter changes the ownership of the object to that of the logged in user (if the current owner is not the logged in user) and performs the operation.



While populating the `wrapperClass` field in the input parameter, give the package information along with the wrapper class name.

For example: `com.tibco.adapter.Clarify.wrapper.Case_Wrapper`

Customized Operations

Customized operations are invoked to perform customized actions on Customized or Extended CBOs.

A customized action is defined to meet specific requirements. The information about the customized CBO should be configured in the project at design time. All the customized methods for a particular CBO are included in its Wrapper Class. For example, the `Case_Wrapper` class contains all customized methods specific to it.

The name of the operation to be performed along with the input values is taken from the incoming request object. Based on the operation name, a customized method is invoked. The result of the operation is then returned to the calling adapter instance after constructing the reply object.

Example of Customized Operation

Given next is a sample schema stored in the project for a customized operation.

- Input Parameters

`MethodName`—Name of the Customized method to be invoked.

`wrapperClass`—Name of the wrapper class

(`com.tibco.adapter.clarify.wrapper.Case_Wrapper`)

`customized_object`—(Object of type any) Map it to the schema of Case Business Object

- Output Parameter

boolean—status of the operation

Table 37 Incoming Message Formats for Customized Operations

Field Name	Type	Default	Field Value Description
MethodName	M_STRING	n/a	Name of the method to be invoked (in the wrapper class of a business object) to perform a customized operation.
wrapperClass	M_STRING	n/a	Name of the wrapper class. For example: <code>com.tibco.adapter..wrapper.Case_Wrapper</code>
customized_object	any	n/a	Schema of the business object on which the customized operation has to be performed. For example: <code>/schema/case</code>

Table 38 Outgoing Message Formats for Customized Operations

Field Name	Type	Default	Field Value Description
returnValue	boolean	n/a	True OR False. Gives the status of the customized operation performed on a customized object.
RPC_SERVER_EXCEPTION	string	n/a	If any exception or exceptions occur while performing a customized operation, the server returns the exception to the calling application.

Custom JavaBean Invocation

The customized operation can also be used to invoke a Custom JavaBean method.

If the `wrapperClass` field in the incoming data starts with `TIBADCLYCUST_`, it is considered as a Custom JavaBean Invocation. The `MethodName` field in the incoming data contains the name of the method to be invoked in the Custom JavaBean.

Ensure that the JavaBean method or the Java class method to be invoked by the adapter has `java.util.Hashtable` and `AppServer` objects as its parameters.

- `AppServer` class contains all the common utility functions used in the adapter and is available with the package `com.tibco.adapter..` Through the `AppServer` object you can access and use the CBO Session created by the adapter.
- The message sent to the adapter is parsed and sent to the invoked method as a `java.util.Hashtable` object. This hashtable retains the hierarchy structure of the incoming message. And the hashtable maps keys to values as defined in the incoming message.

Within the custom JavaBean, you can iterate through the hashtable to access the various fields of the incoming data. The invoked method should return `true` for success and `false` in case of failure.

See [Example of Custom JavaBean Invocation on page 84](#) and [Applying the Custom JavaBean on page 85](#) for more information about the custom JavaBean invocation.

Chapter 8 **Request-Response Invocation Service Functionality**

This chapter describes how the adapter's Request-Response Invocation Service can be used in the Flexible and Non-Flexible deployments.

Topics

- [Request-Response Invocation Service Overview, page 100](#)
- [How the Request-Response Invocation Service Works, page 102](#)

Request-Response Invocation Service Overview

The Request-Response Invocation Service allows users to request data from external applications through the TIBCO environment.

The adapter supports the following approaches:

- [Request-Response Invocation Service Using Flexible Deployment](#)
- [Request-Response Invocation Service Using Non-Flexible Deployment](#)

Request-Response Invocation Service Using Flexible Deployment

The flexible deployment of the Amdocs CRM system entails the business functionality being implemented in separate servers and the communication between the client and the server taking place over BEA Tuxedo.

BEA Tuxedo is a transaction middleware for building reliable distributed systems. BEA Tuxedo can be used to deploy and manage traditional systems independent of the underlying communication in a distributed environment. The adapter uses the BEA Tuxedo middleware to invoke a Request-Response Invocation Service.

In the Flexible Deployment approach, the following components are used:

- **ClearBasic Module**—passes the request data from the ClearBasic clients to the BEA Tuxedo middleware.

The global module uses the Service Message object to make a request FML (Field Manipulation Language) buffer and invoke a synchronous call to the Tuxedo service (Adapter Agent). The global module reads the reply FML buffer and passes it back to the calling ClearBasic forms.

- **Adapter Agent**—which runs as a Tuxedo service, receives the request from the ClearBasic clients as a FML buffer.

It parses this request, constructs a TIBCO Rendezvous Message, and sends it as a Rendezvous request to the adapter. On receiving the reply Rendezvous message from the adapter, the Adapter Agent parses it and writes it to the reply FML buffer, which goes back to the respective Amdocs client. In case the adapter is not available it returns an operation timed out error to the user in the reply FML buffer.

- **TIBCO ActiveMatrix Adapter for Amdocs CRM**—receives the request (TIBCO Rendezvous Message or JMS Message) from the Adapter Agent, after converting it to the configured schema sends it out as an AE request to external applications.

On receiving the reply from an external RPC Server, the adapter sends it back to the agent as a reply Rendezvous message.

Request-Response Invocation Service Using Non-Flexible Deployment

In the Non-flexible deployment scenario the Amdocs ClearBasic client calls a ClearBasic global module, which in turn makes use of the TIBCO Rendezvous ActiveX component to invoke an RPC Client request through the adapter.

Following components are used in this approach:

- ClearBasic Module—passes the request data from the ClearBasic clients to the TIBCO Rendezvous ActiveX component or to the Adapter for Amdocs CRM EMS ActiveX component.

The global module parses the request data, uses the TIBCO Rendezvous ActiveX component to make a request Rendezvous message and sends it to the adapter. The Rendezvous message sent by the ActiveX component consists of sequence of name value pairs of the CBO contained in the form. In addition, it also contains the operation name. The response received also has the same structure as the request sent by the ActiveX component, if all the operations are successful. In case of errors, the error description is passed back to the form as a Rendezvous message. On receiving the reply Rendezvous message, the ClearBasic Module passes it back to the calling ClearBasic forms.

- TIBCO ActiveMatrix Adapter for Amdocs CRM—receives the request Rendezvous Message from the ClearBasic Module, after converting it to the configured schema sends it out as an AE request to external applications.

On receiving the reply from an external RPC Server, the adapter sends it back to the agent as a reply Rendezvous message.

How the Request-Response Invocation Service Works

The Request-Response Invocation Service works with the following forms:

- [ClearBasic Forms, page 102](#)
- [Web Forms, page 109](#)

ClearBasic Forms

Using ClearBasic forms, the Request-Response Invocation Service functions in two modes:

- [Flexible Deployment, page 102](#)
- [Non-Flexible Deployment, page 105](#)

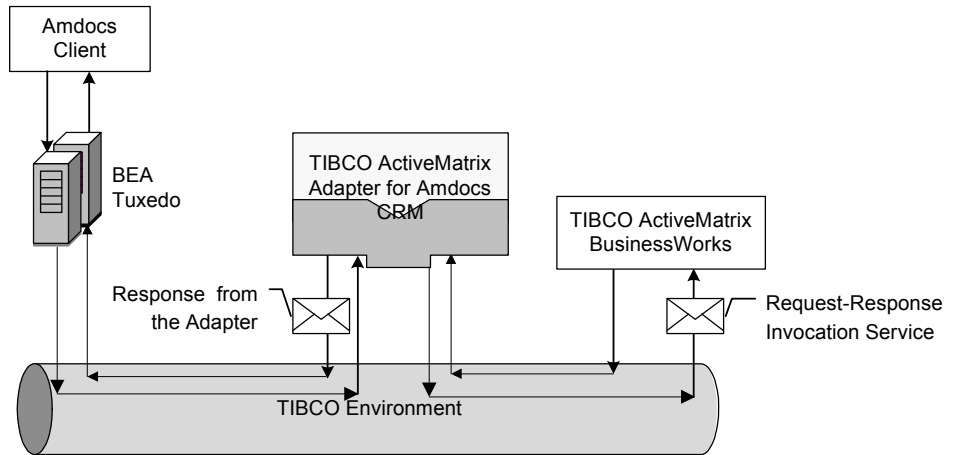
Flexible Deployment

The operations involved in a Flexible deployment are:

1. The client implements the Request-Response Invocation request.

The sample ClearBasic code provided with the adapter installation can be used as a reference, and the Amdocs ClearBasic forms can be modified according to the requirement. See [Sample Code used for a Flexible Deployment on page 104](#) for details.
2. The request is sent to the BEA Tuxedo application, which in turn sends the request to the TIBCO environment.
3. The adapter receives the request, does the necessary processing and sends the response back to the TIBCO environment.
4. The BEA Tuxedo application picks up the response from the TIBCO environment and sends the response to the requesting client.

Figure 16 Flexible Deployment Mode



Prerequisites for Setting Up Request-Response Invocation Functionality using Amdocs ClearBasic Forms in a Flexible Deployment

1. Install and set up Amdocs CRM LAN client. See *Client Installation Guide* for details.
2. Install and set up the adapter. The files and objects needed for Request-Response Invocation Service implementation for LAN client are installed with the adapter.
3. Import the global modules needed for the Request-Response Invocation functionality using the `cbex` tool. See [Enable Publication Service for LAN Client on page 60](#) for details.
4. Install the BEA Tuxedo application.
5. Install the flexible deployment server and set up all the environment variables. See *Flexible Deployment Guide of Amdocs CRM* for details.
6. Copy the Adapter Agent file from the `TIBCO_HOME\adapter\adclycrm\version_num\setup` directory to the `AMDOCSCRM_HOME\appsrv` directory.
7. Set up the BEA Tuxedo service required for the adapter to work.
 - a. Generate the `tuxconfig` file using the sample `ubbconfig` file in the `TIBCO_HOME\adapter\adclycrm\version_num\samples\testRPCClient` directory. Navigate to the `AMDOCSCRM_HOME\appsrv` directory in a command window and run the command:


```
tmloadcf -y ubbconfig
```
 - b. Copy the `fld.tbl` and `tibems.dll` files from the `TIBCO_HOME\adapter\adclycrm\version_num\setup` directory to the `AMDOCSCRM_HOME\appsrv\remotecb_demo` directory.

8. Install TIBCO Rendezvous on the Client machine. See the TIBCO Rendezvous documentation for details.
9. Set the following parameters in the `clarify.env` located in the `AMDOCSCRM_HOME\appsrv` directory:

`login_name`—login user name, such as `sa`

`db_password`—login password, such as `sa`

`db_server`—Amdocs CRM Server host machine

`db_name`—Amdocs CRM Database

10. Use the sample code provided with the adapter installation to configure the form to perform the Request-Response Invocation functionality. See below for details.

Sample Code used for a Flexible Deployment

A sample ClearBasic code for the implementation of Request-Response Invocation functionality is provided with the adapter installation. The sample and its location are given next.

TIBCO_HOME/adapter/adclycrm/*version_num*/samples/testRPCClient
Option Explicit

'The Data type of the rvmessage that is sent to the adapter.

```
Type Reply_Message_UDT
object_name As String
operation_name As String
field_name(500) As String
field_value(500) As String
error_description As String
End Type
```

'The following declarations are made for the methods in Global Module

```
Declare Sub tibcoSetMessageFields(rec As Record, ByVal opName As String, ByVal objName As String, _
ByVal service As String, ByVal network As String, _
ByVal daemon As String, ByVal timeout As Long, _
ByVal subject As String, _
ByRef reply_message As Reply_Message_UDT _
)
```

```
Declare Sub tibcoSetTuxedoFields(rec As Record, ByVal serviceFlag As String, ByVal opName As String, ByVal objName As String, _
ByVal service As String, ByVal network As String, _
ByVal daemon As String, ByVal timeout As Long, _
ByVal subject As String, _
ByRef reply_message As Reply_Message_UDT _
)
```

'in this Sample form a button has been added to the Form and on click of this button(Flexible) a TIBCO Rendezvous message is sent to adapter

```
Sub Flexible_Click()
Dim count As Integer
Dim siteRec As New Record
Set siteRec = Cobj_Location.Contents
Dim reply_message As Reply_Message_UDT
    Call tibcoSetTuxedoFields(siteRec, "JMS", "update", "location", ""7222",_
                                                                    "127.0.0.1", "", 5000, "TEST.CLIENT", reply_message)
```

'if the reply is sent back from the adapter without any errors proceed further

```
If (reply_message.error_description="") Then
Dim i As Integer
i=0
While reply_message.field_name(i) <> "objid"
i=i+1
Wend
SITE_ID.Value= cstr(count) + reply_message.field_value(i)
Else
msgbox(reply_message.error_description)
End If

End Sub
```

Start or Stop the Amdocs CRM Flexible Deployment Application Sever with Tuxedo Services

To start or stop the Amdocs CRM Flexible Deployment application server, follow these steps:

1. Log in to the Windows server as an administrator.
2. Start the Amdocs CRM Flexible Deployment application server from the command line by running:

```
tmboot-y
```

Stop the Amdocs CRM Flexible Deployment application server from the command line by running:

```
tmshutdown-y
```

Non-Flexible Deployment

The operations involved in a Non-Flexible deployment are:

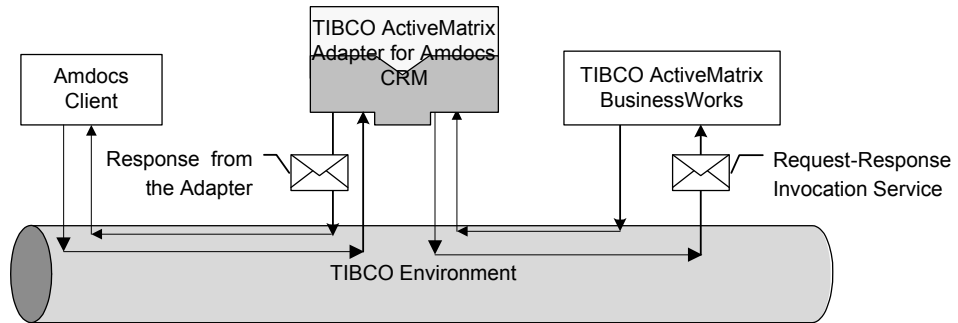
1. The client implements the Request-Response Invocation request.

The sample ClearBasic code provided with the installation can be used as reference, and the Amdocs ClearBasic forms can be modified according to the requirement. See

[Sample Code used for a Non-Flexible Deployment over TIBCO Rendezvous on page 107](#) for details.

2. The request is sent to the TIBCO environment.
3. The adapter receives the request, does the necessary processing and sends the response back to the TIBCO environment.
4. The requesting client receives the response from the TIBCO environment.

Figure 17 Non-Flexible Deployment Mode



Prerequisites for Setting Up Request-Response Invocation Functionality Using Amdocs ClearBasic Forms in a Non-Flexible Deployment

1. Install and set up Amdocs CRM LAN client. See *Client Installation Guide* for details.
2. Install and set up the adapter.

The files and objects needed for Request-Response Invocation Service implementation for LAN client are installed with the adapter.

3. Import the global modules needed for the Request-Response Invocation functionality using the cbex tool. See [Enable Publication Service for LAN Client on page 60](#) for details.
4. To support TIBCO Rendezvous, install TIBCO Rendezvous on the Client machine. See the TIBCO Rendezvous documentation for details.
5. To support JMS, install TIBCO EMS on the Client machine. See the TIBCO EMS documentation for details.
6. If you have opted for JMS support, do the following:

Copy the TIBCOEMS.dll to a folder where you wish to save the COM component.

Execute `tib_reg_emscom.bat` from the `TIBCO_HOME`. Use the following syntax to register the COM component:

```
tib_reg_emscom.bat Path for TIBCOEMS.dll
```

7. Use the sample code provided with the adapter installation to configure the form to perform the Request-Response Invocation functionality.

For TIBCO Rendezvous support, see [Sample Code used for a Non-Flexible Deployment over TIBCO Rendezvous on page 107](#) for details. For JMS support, see [Sample Code used for a Non-Flexible Deployment over JMS on page 108](#) for details.

Sample Code used for a Non-Flexible Deployment over TIBCO Rendezvous

A sample ClearBasic code for the implementation of Request-Response Invocation functionality over TIBCO Rendezvous is provided with the adapter installation. The sample code and its location are given next.

TIBCO_HOME/adapter/adclycrm/version_num/samples/testRPCClient
Option Explicit

'The Data type of the rvmessage that is sent to the adapter.

```
Type Reply_Message_UDT
object_name As String
operation_name As String
field_name(500) As String
field_value(500) As String
error_description As String
End Type
```

'The following declarations are made for the methods in Global Module.

```
Declare Sub tibcoSetMessageFields(rec As Record, ByVal opName As String, ByVal objName As String, _
    ByVal service As String, ByVal network As String, _
    ByVal daemon As String, ByVal timeout As Long, _
    ByVal subject As String, _
    ByVal reply_message As Reply_Message_UDT _
)
```

```
Declare Sub tibcoSetTuxedoFields(rec As Record, ByVal opName As String, ByVal objName As String, _
    ByVal service As String, ByVal network As String, _
    ByVal daemon As String, ByVal timeout As Long, _
    ByVal subject As String, _
    ByVal reply_message As Reply_Message_UDT _
)
```

'In this Sample form a button has been added to the Form and on clicking the button (NonFlexible) a TIBCO Rendezvous message is sent to the adapter.

```
Sub NonFlexible_Click()
Dim siteRec As New Record
Set siteRec = Cobj_Location.Contents
Dim reply_message As Reply_Message_UDT
```

'Call the tibcoSetMessageFields with the Record variable, operation name, service,network,daemon,wait time.

```
Call tibcoSetMessageFields(siteRec, "update", "location", "7500", _
```

```

    "", "tcp:7500", 100, "TEST.CLIENT", reply_message)
If (reply_message.error_description="") Then
Dim i As Integer
i=0
While reply_message.field_name(i) <> "objid"
i=i+1
Wend
SITE_ID.Value= "" + reply_message.field_value(i)
Else
msgbox(reply_message.error_description)
End If

End Sub

```

Sample Code used for a Non-Flexible Deployment over JMS

A sample ClearBasic code for the implementation of Request-Response Invocation functionality over JMS is provided with the adapter installation. The sample code and its location are given next.

TIBCO_HOME/adapter/adclycrm/*version_num*/samples/testRPCClient
Option Explicit

'The Data type of the JMSMessage that is sent to the adapter.

```

Type Reply_Message_UDT
object_name As String
operation_name As String
field_name(500) As String
field_value(500) As String
error_description As String
End Type

```

'The following declarations are made for the methods in Global Module.

```

Declare Sub tibcoSetMessageFieldsEMS(rec As Record, ByVal opName As String, ByVal objName As String, _
ByVal service As String, ByVal network As String, _
ByVal daemon As String, ByVal timeout As Long, _
ByVal subject As String, _
ByRef reply_message As Reply_Message_UDT _
)

```

```

Declare Sub tibcoSetTuxedoFields(rec As Record, ByVal opName As String, ByVal objName As String, _
ByVal service As String, ByVal network As String, _
ByVal daemon As String, ByVal timeout As Long, _
ByVal subject As String, _
ByRef reply_message As Reply_Message_UDT _
)

```

'In this Sample form a button has been added to the Form and on clicking the button(NonFlexible) a JMS message is sent to the adapter.

```

Sub NonFlexibleJMS_Click()
Dim siteRec As New Record

```



```
Set siteRec = Cobj_Location.Contents
Dim reply_message As Reply_Message_UDT
```

'call the tibcoSetMessageFieldsEMS with the Record variable ,operation name, service,network,daemon,wait time.

```
Call tibcoSetMessageFieldsEMS(siteRec, "update", "location", "tcp://10.97.97.55:7222", _
    "", "", 5000, "TEST.CLIENT", reply_message)
If (reply_message.error_description="") Then
Dim i As Integer
i=0
While reply_message.field_name(i) <> "objid"
i=i+1
Wend
SITE_ID.Value= "SiteID:" + reply_message.field_value(i+1)
Else
msgboreply_message.error_description)
End If

End Sub
```

Web Forms

Using web forms, the Request-Response Invocation Service functions in the Web Client deployment mode.



The adapter supports the following web applications:

- Amdocs CRM eOrder
- Amdocs CRM eSupport

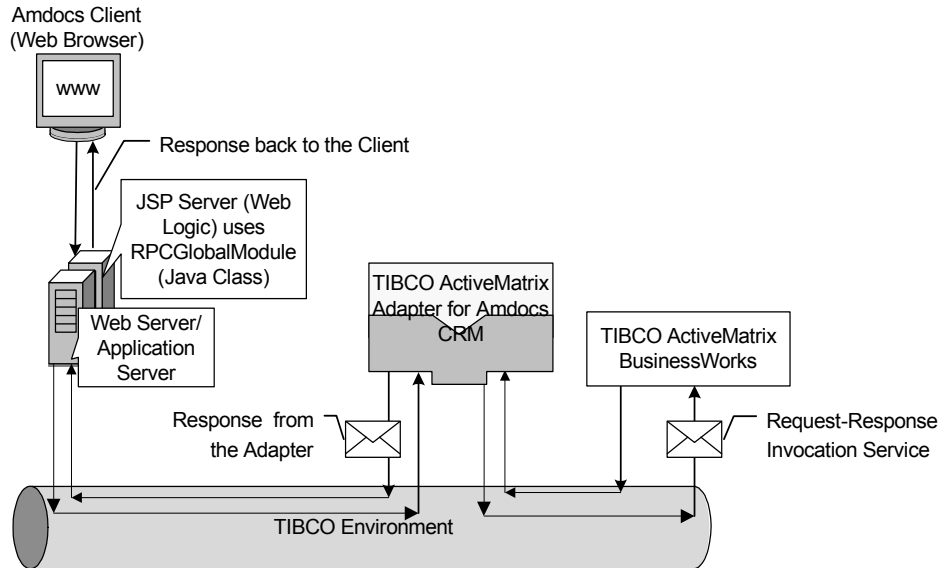
The adapter uses the Java Class Module, `RPCGlobalModule.jar`, to invoke a Request-Response Invocation Service. The Amdocs web applications use BEA Weblogic server. The sample implementation (provided with the adapter) uses Amdocs objects, JSP pages and the `RPCGlobalmodule` class file.

The operations involved in a Web Client deployment are:

1. The Amdocs CRM client submits the JSP form to the Web server/JSP Server.
2. The web server processes the page and executes the Java code in the requested page. The `RPCGlobalModule` Class method is called in Java code, which sends the request message to the adapter via the TIBCO environment.
3. The adapter receives the request, does the necessary processing and sends the response back to the web server via the TIBCO environment.
4. The requesting client receives the response from the web server.

The sample JSP code provided with the installation can be used as reference, and the JSP forms can be modified according to the requirement. See [Sample Code Used for Web Forms on page 112](#) for details.

Figure 18 Web client deployment mode



Prerequisites for Setting Up Request-Response Invocation Functionality Using Amdocs Web Forms

All Amdocs web applications use BEA Weblogic server. Carry out the following steps to set up RPC functionality in the Amdocs web forms:

1. Install and set up Amdocs CRM Web client. See *Client Installation Guide* for details.
2. Install and set up the adapter. The files and objects needed for RPC client implementation for web forms are installed with the adapter.
3. Install and set up eOrder web application, so that you can customize the JSP pages provided with the installation as per the sample code provided with the adapter installation. See [Sample Code Used for Web Forms on page 112](#).
4. Install and set up the BEA Weblogic server. See *Client Installation Guide* for details.
5. Install TIBCO Runtime Agent on the machine where BEA Weblogic is installed. TIBCO Runtime Agent supports both RV and JMS libraries. See the TIBCO Rendezvous documentation for details.

- 6. Add the `RPCGlobalModule.class` and CBOs to BEA Weblogic CLASSPATH environment variable. See the BEA Weblogic documentation for details and information on security.
- 7. Ensure that the location of the Global module classpath is included in the BEA Weblogic startup script.
- 8. Add location of CBOs and `RPCGlobalModule` class to the CLASSPATH in your BEA Weblogic startup script.

Programmer Reference to Use the `RPC GlobalModule`

Class Name	<code>RPCGlobalModule</code>
Function Name	<code>TibcoSetMessageFields()</code>
Declaration	<pre>tibcoSetMessageFields (com..cbo base, java.lang.String Operation, java.lang.String ObjectName, java.lang.String Service, java.lang.String Network, java.lang.String Daemon, double Timeout, java.lang.String SubjectName, java.util.Vector CurMsgNum) throws TibrvException</pre>
Purpose	Returns the RPC Client response in the vector parameter being passed.

Table 39 Parameter List of Function `TibcoSetMessageFields()`

Parameter	Description
Base	Amdocs object.
Operation	Pass the type of operation to be executed.
ObjectName	Object being passed.
Service	RVD service parameter.
Network	RVD network parameter.
Daemon	RVD Daemon parameter.
Timeout	Value should be greater than the Amdocs adapter response time for RPC requests.
SubjectName	RVD subject name parameter.

Table 39 Parameter List of Function *TibcoSetMessageFields()*

Parameter	Description
CurMsgNum	Holds the return value of the request response call.

Function Name	TibcoSetMessageFieldsEMS()
Declaration	<div>TibcoSetMessageFieldsEMS (com..cbo base, java.lang.String Operation, java.lang.String ObjectName, java.lang.String Service, java.lang.String Network, java.lang.String Daemon, double Timeout, java.lang.String SubjectName, java.util.Vector CurMsgNum) throws TibrvException</div>
Purpose	Returns the RPC Client response in the vector parameter being passed.

Table 40 Parameter list of function *TibcoSetMessageFieldsEMS()*

Parameter	Description
Base	Amdocs object.
Operation	Pass the type of operation to be executed.
ObjectName	Object being passed.
Service	EMS server, port, and address.
Network	N/A
Daemon	N/A
Timeout	Value should be greater than the Amdocs adapter response time for RPC requests.
SubjectName	Subject Name.
CurMsgNum	Holds the return value of the request response call.

Sample Code Used for Web Forms

A sample JSP page for the implementation of Request-Response Invocation functionality is provided with the adapter installation. The location of the sample code is given next.

TIBCO_HOME/adapter/adclycrm/*version_num*/samples/testRPCClient

The `RPCGlobalModule` interface `tibcoSetMessageFields` enables the RPC client functionality for the web forms. The sample JSP code demonstrates the usage of the functionality and the changes to be made to the `ordersubmit.jsp` page that is installed as part of Amdocs CRM web client installation.

```

.....
boOEQuote.getFields().getItem("total_tax_amt").setValue(boOeQuote2.getFields().getItem("total_tax_amt").getValue(
).toString());
boOEQuote.getFields().getItem("tot_ship_amt").setValue(boOeQuote2.getFields().getItem("tot_ship_amt").getValue().
toString());
boOEQuote.getFields().getItem("handling_cost").setValue(boOeQuote2.getFields().getItem("handling_cost").getValue(
).toString());
boOEQuote.getFields().getItem("total_net").setValue(boOeQuote2.getFields().getItem("total_net").getValue().toString(
));
boOEQuote.submit();
boOEQuote.update();
//CUSTOMIZED CODE BEGINS HERE.....
try {
    Vector currMsgNum = new Vector();
    RPCGlobalModule tc= new RPCGlobalModule();
    tc.tibcoSetMessageFields(boOEQuote,"select","R1","","",100,"TEST.CLIENT", currMsgNum);
}
catch(Exception tibException) {
    System.out.println(tibException);
    errorMsg.setDefaultStatusTitle("TIBCO_ORDERSUBMIT_ERR");
    errorMsg.setSpecialStatusMessage(ClfyForm.getError().getDescription() + "\r\n" + tibException.toString() );
    response.sendRedirect("../CommonJSP/ErrorPage.jsp");
}
//CUSTOMIZED CODE ENDS HERE

eOrderCommon.SendRoutingRequest(boOEQuote);
pageInit.setAttributeBo("shoppingcart", boOeQuote2 , ClfyForm, session);

```


Chapter 9

Deploying and Starting an Adapter Using TIBCO Administrator

This chapter provides an overview about deploying, starting, stopping, and monitoring adapter services using TIBCO Administrator.

Topics

- [Creating an EAR File in TIBCO Designer, page 116](#)
- [Deploying the Project, page 117](#)
- [Starting or Stopping the Adapter, page 119](#)
- [Monitoring the Adapter, page 120](#)

Creating an EAR File in TIBCO Designer

The Enterprise Archive file (EAR) contains information on what you want to deploy. This could be one or more adapter services, one or more TIBCO ActiveMatrix 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.

To create an EAR file in TIBCO Designer, follow these steps:

1. Configure the adapter services.
2. Select the project in the Project Panel.
3. Drag the **Enterprise Archive** icon from the General Palettes Panel to the Design Panel.
4. Select the Enterprise Archive you have just created in the Project Panel.

If there are any configured adapter services in your project, an Adapter Archive resource becomes available in the Palettes panel.

5. Drag the **Adapter Archive** icon from the Palettes panel to the Design Pane.
6. Configure the adapter archive in the Configuration Panel. Click the **Browse resources** button to select the adapter instance in the Adapter field and then click the **Apply** button.
7. Select the Enterprise Archive in the Project Panel, and then click the **Build Archive** button in the Configuration Panel to create the archive file.

See Also

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

Deploying 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) can be 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. Import the EAR file into TIBCO Administrator.
 - a. Start TIBCO Administrator. Log into the domain.
 - b. In the left panel, select **Application Management**.
 - c. Click the **New Application** button.
 - d. In the Upload EAR File dialog, browse to select the EAR file. Click the **OK** button.
2. Assign adapter archives in the EAR file to adapters installed in the administration domain and likewise assign process archives to process engines.
3. In the New Application Configuration dialog, uncheck the **Quick Configure** checkbox, and then click the **Save** button.
4. Specify startup options for each adapter service.
5. In the Configuration Builder, click the top level application name, and then perform following tasks:
 - a. Change the values in the Advanced tab if necessary.
 - b. Click the **Save** button.
 - c. In the Configuration Builder, select the adapter archive, and then click the **Add to Additional Machines** button.
 - d. Select the machine and click the **OK** button.
 - e. Click the **Save** button.
6. In the Configuration tab, click the **Deploy** button, and then click the **OK** button.
7. Once the deployment is complete, click the **Application Processes** button under the application. Select the adapter and the TIBCO ActiveMatrix BusinessWorks engine and click the **Start Selected** button.

See Also

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

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

Starting or Stopping the Adapter

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

Starting the Adapter

To start an adapter service from the module:

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

The status changes from Stopped to Starting up to Started.

Stopping the Adapter

To stop the adapter service, click the **Stop Selected** button.

See Also

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

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

Chapter 10 **Configuring Advanced Features**

This chapter describes advanced configuration options available in the palette.

Topics

- [Using the Adapter with a Revision Control System, page 122](#)
- [Modifying the DTA Rendezvous Connection Properties, page 124](#)
- [Defining a TIBCO Hawk Session, page 125](#)
- [Using Global Variables, page 126](#)
- [Configuring a Remote Adapter, page 132](#)
- [Using the Tracking Element, page 133](#)

Using the Adapter with a Revision Control System

TIBCO Designer supports revision control systems such as MicroSoft Visual SourceSafe and Perforce. To use a revision control system, you must manually add some configured resources to the revision control system and check in the resources when completing the instance configuration.

As part of instance configuration, the adapter creates certain schema files. For example, when you drag an instance *Instance1*, the following files and folders are created:

```
Project_root/AESchemas/ae/.aeschema
Project_root/AESchemas/ae//Instance1
```

Next, as a part of the service configuration, the adapter creates schema files in:

```
Project_root/AESchemas/ae//Instance1
```

For example, when you configure a Subscription service, *Sub1*, the following file is created:

```
Project_root/AESchemas/ae//Instance1/Sub1.aeschema
```

Certain sequences and classes are added in:

```
Project_root/AESchemas/ae/.aeschema
```

When the project is saved and a revision control system specified, the adapter displays a warning that additional files were created and should be added to the revision control system. This warning appears only when the files are created for the first time.

The warning displays a **Go To Resource** button that helps in navigating to the resource.

You should use the *Multi-User>Add Resources to RCS* menu command to add these files to the revision control system. For information about how to use the Multi-User feature, see *TIBCO Designer User's Guide*.

Copy, Cut, Paste and Move Operations

- To successfully copy and paste a service from adapter *Instance1* to *Instance2*, the adapter configuration and schema files for the *Instance2* must be checked out.
- To successfully cut and paste a service from adapter *Instance1* to *Instance2*, the adapter configuration and schema files for both *Instance1* and *Instance2* must be checked out.
- To successfully move a service from adapter *Instance1* to *Instance2*, the adapter configuration and schema files for both *Instance1* and *Instance2* must be checked out.

Regeneration When Moving, Copying and Pasting

- Default subjects are not regenerated to reflect the new instance name when a service is moved.

- Manually changed certified messaging and certified messaging queue ledger file names are regenerated to defaults when a service is moved, or copied and pasted to a new instance.
- If a service associated with a custom session is moved, or copied and pasted, the custom session is not moved, or copied and pasted. The session is regenerated as a default session.

Creating Two Adapter Instances in Same Project by Different Users

Use the following steps as guidelines when creating two adapter instances in the same project by two users:

1. Create a new project, myTestProj.
2. Save the project using a revision control system, for example, MicroSoft Visual Source Safe.
3. Click the **root** folder and add the files to the revision control system. Click the **yes** button for recursive addition.
4. Check in the project.
5. Sync the project on two machines. Machine A and Machine B.
6. Open the project on Machine A and configure Instance1. Add the Instance1** and the related schema files to the revision control system. Check in the changes.
7. Open the project on Machine B and configure Instance2. Add the Instance2** and the related schema files to the revision control system. Check in the Changes.



The global variables must be populated first, otherwise, you cannot configure the instances.

Modifying the DTA Rendezvous Connection Properties

You can modify the TIBCO Rendezvous connection properties used by the design-time adapter in the `adclyCRMDTA_RV.tra` properties file or in the `adclyCRMDTA.dat` project. Both files are located in the `bin` directory.

Modifying the Properties File

The `adclyCRMDTA.tra` properties file contains properties that specify, for example, the pathname to the project where configuration information for the design-time adapter is defined, the location of the directories where trace files and the certified messaging ledger files are stored, and so on. Properties specified in the `adclyCRMDTA_RV.tra` properties file overwrite the same properties specified in the project.

To specify custom Rendezvous connection properties for the design-time adapter:

1. Change directory to the `bin` directory. For example:

```
TIBCO_HOME/adapter/adclycrm/version_num/bin
```

2. Open the `adclyCRMDTA_RV.tra` properties file with a text editor.
3. Add or edit the following properties:

```
tibco.clientVar.RvDaemo value
```

```
tibco.clientVar.RvNetwork value
```

```
tibco.clientVar.RvHost value
```

where *value* is a valid TIBCO Rendezvous daemon, network or host value. See the TIBCO Rendezvous documentation for correct values to use.

4. Restart the design-time adapter to make these modifications take effect.

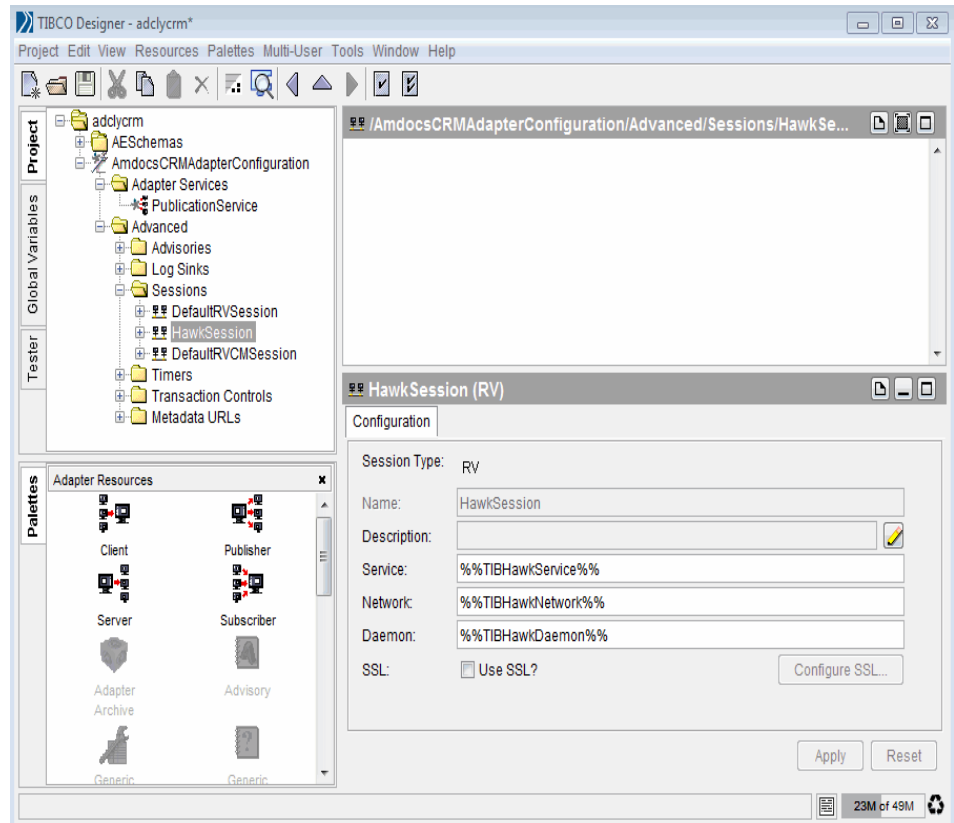
Defining a TIBCO Hawk Session

A default TIBCO Rendezvous session, *HawkSession*, is defined in the project whenever a new instance of the adapter is configured. You can use this session to monitor the adapter using TIBCO Hawk.

To modify the parameters of *HawkSession*:

1. In the project tree panel, click the Adapter Configuration button defined for your adapter instance.
2. Select **Advanced > Sessions > HawkSession**.
3. The default values in the Service, Network and Daemon fields are 7474, None and tcp:7474. Change the default values if required and click the **Apply** button.

Figure 19 Hawk session



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 contain no white space.

Variable substitution allows you to:

- 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 129](#).

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

Specifying Global Variables Using TIBCO Designer

Global variables provide an easy way to set defaults for use throughout your project. There are several ways to use them:

- Define a variable using TIBCO Designer, then override the value for individual applications at deployment time using TIBCO Administrator. You can also override values for predefined variables, unless the GUI does not allow you to set them later.
- Predefine a variable using TIBCO Designer, then override the value for individual services (for example, Publication service or TIBCO ActiveMatrix BusinessWorks process) at deployment time using TIBCO Administrator. The values are then used at runtime. You can also override values for predefined variables, unless the GUI does not allow you to set them later.

For example, you could assign the value 7474 to the predefined global variable `RvDaemon`. Then use the variable in different sessions in your adapter. 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.

Using Global Variables in the Project

To use global variables in your project:

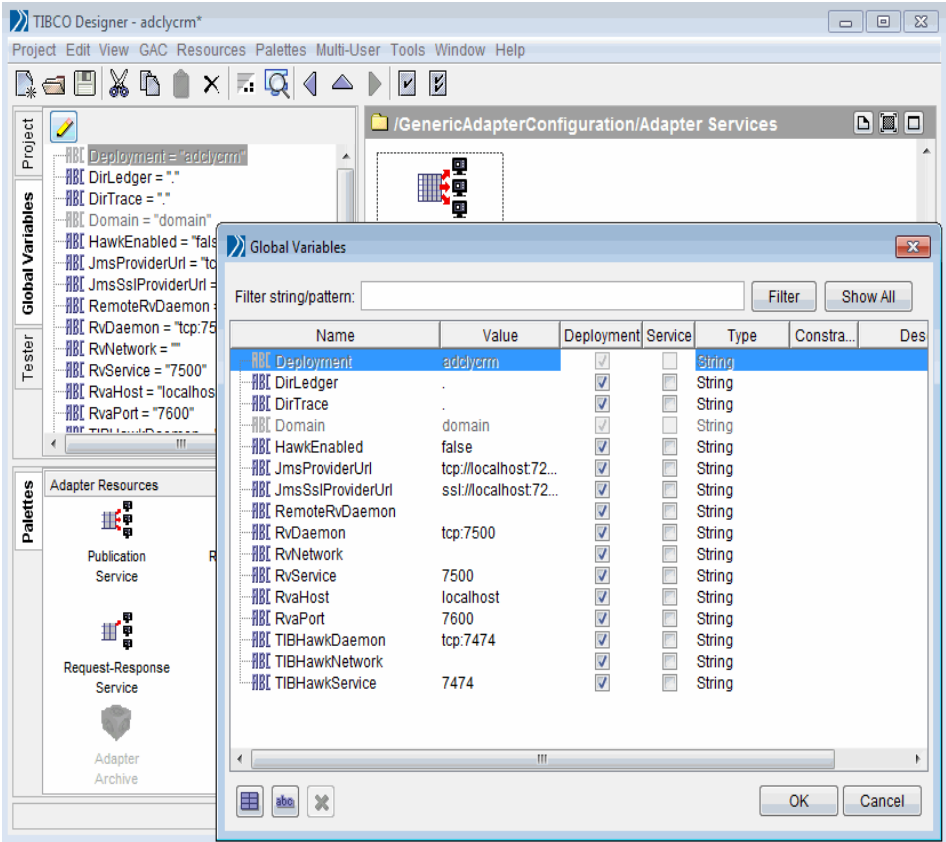
1. In the project panel, select the **Global Variables** tab.

The project panel is updated to display all currently defined global variables. Click the **Open Advanced Editor** button (pencil button at the top left corner). 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 are done.
- To add a new global variable group, click the **leftmost, Add a Variable Group** buttons, at the bottom of the dialog box. 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** button. 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 are done.

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

Figure 20 Global variables



2. 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 running, 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).

A number of global variables are predefined. See [Predefined Global Variables on page 129](#) for information. You may add definitions of any variables you need to the predefined variables.

Changing Global Variable Values at Runtime

You can change the value of a global variable when you deploy your project in TIBCO Administrator. See the section on modifying runtime variables in *TIBCO Administrator User's Guide* for more information.

You can also specify values for global variables when starting a process engine on the command line. To do this, specify the following as a command line argument when starting the process engine:

```
-tibco.clientVar.variablePathAndName value
```

where *variablePathAndName* is the name of the variable you wish to set, including the path to the variable if it is contained in a folder. *value* is the value you wish to set the variable to.

For example, if you have a global variable named `item1` contained in a folder named `myGroup` and you wish to set its value to 500, add the following argument to the command line when starting the process engine:

```
-tibco.clientVar.myGroup/item1 500
```

Predefined Global Variables

[Table 41](#) lists and explains the predefined global variables. Some global variables are automatically used within the system when an adapter instance is configured.

Table 41 Predefined Global Variables (Sheet 1 of 3)

Variable	Description
Deployment	Defaults to the TIBCO Designer project name. 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.
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 <code>MyDomain</code> . 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. <code>True</code> indicates that a Hawk microagent is defined for the adapter. <code>False</code> indicates the microagent is not to be used.
JmsProviderUrl	Specifies where the JMS server is located. Setting this value mostly makes sense in early stages of a project, when only one JMS daemon is used.
JmsSslProviderUrl	Specifies where the JMS SSL daemon is located.

Table 41 Predefined Global Variables (Sheet 2 of 3)

Variable	Description
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 needs 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>
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 7600.
TIBHawkDaemon	TIBCO Rendezvous daemon used in the TIBCO Hawk session. See the <i>TIBCO Hawk Installation and Configuration</i> manual for details.
TIBHawkNetwork	TIBCO Rendezvous network used by the TIBCO Hawk session. See the <i>TIBCO Hawk Installation and Configuration</i> manual for details.
TIBHawkService	TIBCO Rendezvous service used by the TIBCO Hawk session. See <i>TIBCO Hawk Installation and Configuration</i> manual for details.
adclycrm.connection.database	Used by the system to identify theAmdocs database the adapter needs to connect to.
adclycrm.connection.host	Used by the system to identify the Amdocs database host name. Used by the Publisher during runtime only.

Table 41 Predefined Global Variables (Sheet 3 of 3)

Variable	Description
adclycrm.connection.password	Password to be used by the system to log into the Amdocs database.
adclycrm.connection.port	The JDBC port on which the adapter connects to the database.
adclycrm.connection.server	Used by the system to identify the Amdocs database server name.
adclycrm.connection.user	User name or User ID to be used by the adapter to log into the Amdocs database.
adclycrmDTAJmsProviderURL	Used by the system to identify the JMSProviderURL to connect to the design-time adapter.
adclycrmDTARvDaemon	Used by the system to identify the TIBCO Rendezvous daemon parameter to connect to the design-time adapter. The parameter instructs the transport object about how and where to find the Rendezvous daemon and establish communication. See <i>TIBCO Rendezvous Concepts</i> for details.
adclycrmDTARvNetwork	Used by the system to identify the TIBCO Rendezvous network parameter to connect to the design-time adapter. Every network transport communicates with other transports over a single network interface. On computers with more than one network interface, the network parameter instructs the TIBCO Rendezvous daemon to use a particular network for all outbound messages from this transport. See <i>TIBCO Rendezvous Concepts</i> for details.
adclycrmDTARvService	Used by the system to identify the TIBCO Rendezvous service parameter used to connect to the design-time adapter. The Rendezvous daemon divides the network into logical partitions. Each transport communicates on a single service; a transport can communicate only with other transports on the same service. See <i>TIBCO Rendezvous Concepts</i> for details.

Configuring a Remote Adapter

If only TIBCO Designer is installed on your computer and the adapter is installed on a remote computer, you can still configure the adapter from your computer. To do this:

1. In a command prompt window, copy the two adapter palette files (`adcllyCRMpalette.dat` and `adcllyCRMpalette.jar`), located in `TIBCO_HOME/adapter/adcllycrm/version_num/lib/palettes`, from the remote computer to your computer.
2. Go to the `TIBCO_HOME/Designer/version_num/bin` directory and open the `designer.tra` file. Modify the path specified for the property that enables Designer to find the adapter's palette as shown:

```
#application.args -d  
java.property.palettePath  
  
TIBCO_HOME/tibco/adapter/adcllycrm/version_num/lib/palettes
```
3. Save and close the `designer.tra` file.
4. Open the project in TIBCO Designer. This project must be server based. In the project tree panel, select the **AdapterConfiguration** icon and configure it as desired. This icon represents the remote adapter configuration.

Using the Tracking Element

Trace messages contain a tracking element, which is helpful in troubleshooting because it tracks all the components, products, and processes in your installation related to the problem.

The tracking element has the format:

`tracking=#trackingID#infoItem1#infoItem2...[#infoItemn#`

The tracking ID has the format:

`tracking=#IDpart1--IDpart2--IDpart3--IDpart4#`

An example of tracking ID is given below:

2003 May 15 15:09:38:277 GMT +5 Info [Adapter] AE-0000445 "Message to be published :1"

`tracking=#QdUThbTh8n/Uak9u9Azzw6BUzzw#`

Tracking Element ID

The trace message above indicates the adapter published a message. The `#QdUThbTh8n/Uak9u9Azzw6BUzzw#` tracking identifier uniquely identifies the message.

Chapter 11 **Monitoring the Adapter Using TIBCO Hawk**

This chapter explains how to use TIBCO Hawk microagents to monitor and manage the adapter.

Topics

- [Overview, page 136](#)
- [Starting TIBCO Hawk Software, page 137](#)
- [The Auto-Discovery Process, page 138](#)
- [Invoking Microagent Methods, page 139](#)
- [Available Microagents, page 142](#)

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 e-mail, paging, running executables
- 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 following 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**—Feeds information back to TIBCO Hawk and exposes 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 *TIBCO Hawk Installation and Configuration* for details.

The *TIBCO Hawk Administrator's Guide* explains how to start the TIBCO Hawk Display.

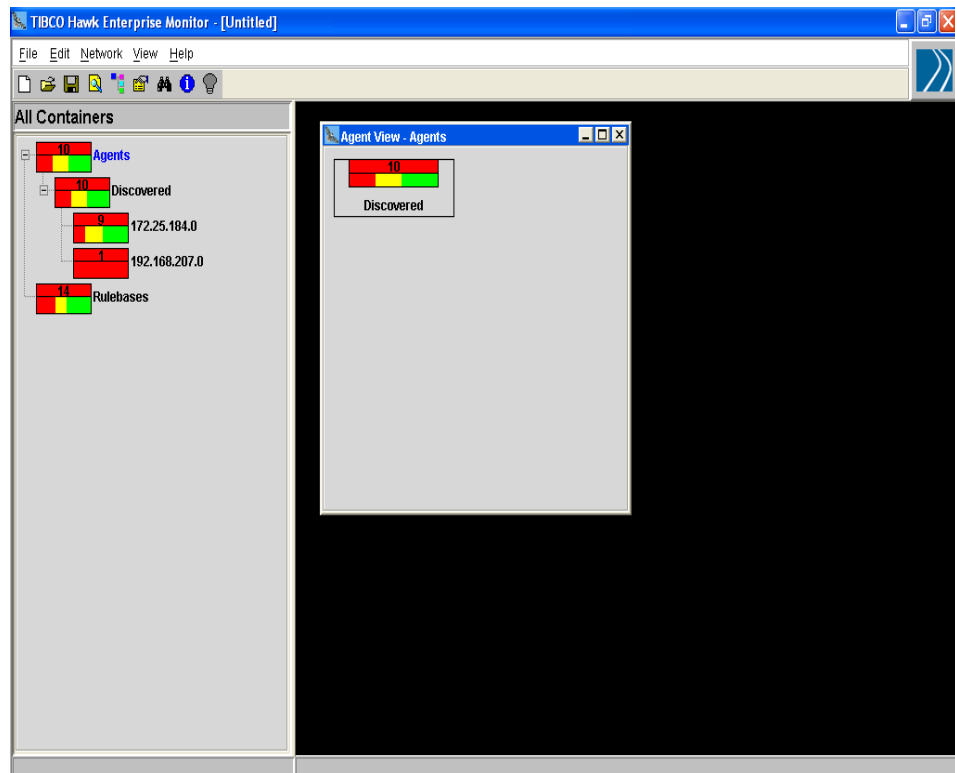
The guides are included in your TIBCO Hawk software installation area.

The Auto-Discovery Process

Once starting 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.

Figure 21 TIBCO Hawk Enterprise Monitor



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 information of icon elements and characteristics, see *TIBCO Hawk Administrator's Guide*.

Invoking Microagent Methods

A set of default microagents, platform-specific and platform-independent, is loaded when a TIBCO Hawk Agent is started. When you install and start the TIBCO ActiveMatrix Adapter for Amdocs CRM, microagents for the adapter are dynamically added to the local agent.

To invoke a microagent method on a TIBCO Hawk Agent:

1. In TIBCO Hawk Display, right-click the **agent** icon and select **Get Microagents**.

If TIBCO Hawk security is implemented on your system and you have no access to microagents on this agent, an error dialog displays. Select another agent, or contact your system administrator to obtain access.

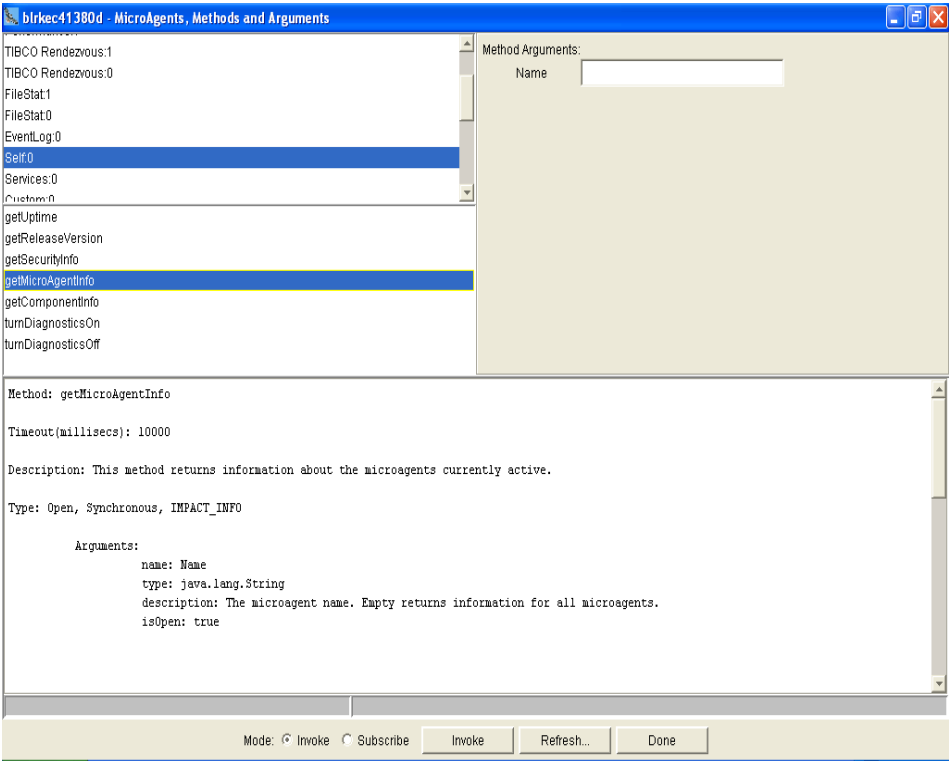
The Microagents, Methods and Arguments dialog displays. The panel on the upper left lists microagents you can access on the current agent. This dialog has two modes, *Invoke* and *Subscribe*:

- 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 `getMicroAgentInfo`.

Figure 22 Select a method in the Microagents, Methods and Arguments dialog



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.

Figure 23 Invocation results dialog

Name	Display Name	Count	Help
COM.TIBCO.hawk.hma.EventLog	EventLog	2	TIBCO Hawk/Windows Event Log Microagent
COM.TIBCO.hawk.hma.Process	Process	2	TIBCO Hawk Process Microagent
COM.TIBCO.hawk.hma.FileStat	FileStat	2	TIBCO Hawk File Status Microagent
COM.TIBCO.hawk.hma.TibRendezvous	TIBCO Rendezvous	2	TIBCO Hawk Rendezvous Microagent
COM.TIBCO.hawk.hma.Services	Services	2	TIBCO Hawk Windows Services Microagent
COM.TIBCO.hawk.microagent.Self	Self	1	TIBCO Hawk built-in Microagent
IM Engine - AE Tracing Hawk Sink for imed_...	IM Engine - AE Tracing Hawk Sink for imed_...	1	Provides trace messages from IM Engine
COM.TIBCO.hawk.microagent.Custom	Custom	1	TIBCO Hawk built-in Microagent
IM Engine Administration for imed_debug_e...	IM Engine Administration for imed_debug_e...	1	HAWK agent for the shell
COM.TIBCO.hawk.hma.Registry	Registry	2	TIBCO Hawk Windows Registry Microagent
COM.TIBCO.hawk.hma.Performance	Performance	2	TIBCO Hawk Windows Performance Microa...
IM Engine Monitor for imed_debug_engine1	IM Engine Monitor for imed_debug_engine1	1	Various Engine monitoring funtions
COM.TIBCO.hawk.microagent.Logfile	Logfile	1	TIBCO Hawk built-in Microagent
COM.TIBCO.hawk.microagent.RuleBaseEn...	RuleBaseEngine	1	TIBCO Hawk built-in MicroAgent
COM.TIBCO.hawk.microagent.SysInfo	SysInfo	1	TIBCO Hawk built-in Microagent

Click a cell to display its value in this area!

Done

7. Click the **Done** button 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 *TIBCO Hawk Administrator's Guide*.

Available Microagents

The adapter has two microagents, a standard TIBCO Hawk microagent named `COM.TIBCO.ADAPTER.xyz` where `xyz` is the adapter configuration name and a class microagent. The microagents provide:

- Business level statistics. Statistics that report the progress of the adapter as it interacts with the vendor application.

For example, in a database adapter such statistics might indicate whether objects were successfully or unsuccessfully inserted, updated, or deleted in the database.

- 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 is getting and setting the polling interval. Updating a runtime parameter through the Hawk microagent only affects the setting of the running instance. It does not make a permanent change of the setting in either the project or the `.tra` file.

By default, all microagents, standard and class microagents are available at runtime.

The `perfMon` property value set in the adapter’s property file affects the business statistics related methods. If this property is set to `on`, the adapter does all the performance related calculations. When you invoke the methods, if the `perfMon` property is set to `off`, default values are displayed and not the valid values.

Table 42 lists each method available for the adapter and page on which the method is explained.

Table 42 Microagent Methods (Sheet 1 of 3)

Method	Description	Page
activateTraceRole()	Activates a mapping of a role to a sink at runtime.	145
deactivateTraceRole()	Deactivates a mapping of roles to a sinks at runtime.	146
getActivityStatistics()	Returns the total number of objects processed for all the schemas.	147

Table 42 Microagent Methods (Sheet 2 of 3)

Method	Description	Page
getActivityStatisticsByOperation(Operation)	Returns the total number of objects processed for all the schemas by each service associated with a specified operation.	148
getActivityStatisticsBySchema(SchemaName)	Returns statistics about any activities on a particular object or schema.	149
getActivityStatisticsByService(ServiceName)	Returns statistics about the data handled by a particular adapter service.	150
getAdapterServiceInformation()	Returns information about the services implemented by this adapter.	151
getComponents()	Returns information about the publisher, subscriber and IODescriptor.	152
getConfig()	Returns basic configuration information. More specific information is accessed by the more specific methods.	153
getConfigProperties()	Returns a list of publishers and subscribers.	154
getConnectionStatistics()	Returns the state and statistics for all the current connections used by the adapter.	155
getHostInformation()	Returns standard and extended application information.	156
getPerfMonSetting()	Returns the setting of the perfMon option.	157
getPollingBatchSize()	Returns the polling batch size.	158
getPollingInterval()	Returns the current polling interval setting.	159
getQueueStatistics()	Returns the current count of elements in any internal queue used by the adapter.	160
getRvConfig()	Returns information about all TIBCO Rendezvous sessions defined.	161
getStatus()	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.	162
getThreadStatistics()	Returns the operation counts of current threads.	163

Table 42 *Microagent Methods (Sheet 3 of 3)*

Method	Description	Page
getTraceSinks()	Returns information about sinks to which traces currently go.	164
getVersion()	Returns the instance ID, application name, version, and date for this adapter instance.	165
_onUnsolicitedMsg()	Displays alert messages sent to the current adapter.	166
preRegisterListener()	Preregisters an anticipated listener.	167
resetActivityStatistics()	Resets all the counts for the activity statistics.	168
resetConnectionStatistics()	Resets all the counts for the connection statistics.	169
resetThreadStatistics()	Resets all the counts for the thread statistics.	170
reviewLedger()	Returns information retrieved from the ledger file of a certified messaging session for a publisher adapter.	171
setPollingBatchSize()	Sets the polling batch size.	172
setPollingInterval()	Sets the polling interval for the Publication service.	173
setTraceSinks()	Adds a role or changes the file limit of a previously specified sink.	174
stopApplicationInstance()	Stops the running adapter instance.	175
unRegisterListener()	Unregisters a preregistered listener.	176

activateTraceRole()

Activates a mapping of a role to a sink at runtime. This replaces the now deprecated `setTraceSink()` Hawk method.

Table 43 *activateTraceRole()*

Parameters	Type	Description
roleName	string	Name of the role to activate.
sinkName	string	Name of a sink for which to activate the role.

deactivateTraceRole()

Deactivates a mapping roles to a sinks at runtime.

Table 44 deactivateTraceRole()

Parameters	Type	Description
roleName	string	Name of the role to deactivate.
sinkName	string	Name of a sink for which to deactivate the role.

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.

Table 45 Input parameter of getActivityStatistics()

Input Parameter	Type	Description
GetSubTotalBy	string	Indicates how to group the subtotals, by Service Or Operation.

Table 46 Returns of getActivityStatistics()

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.

getActivityStatisticsByOperation(Operation)

Returns statistics about one operation.

Table 47 Input parameter of getActivityStatisticsByOperation(Operation)

Input Parameter	Type	Description
Operation	string	Name of the operation.

Table 48 Returns of getActivityStatisticsByOperation(Operation)

Returns	Type	Description
Operation	string	Name of the operation.
Service Name	string	Name of the service.
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.

getActivityStatisticsBySchema(SchemaName)

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.

Table 49 Input parameter of getActivityStatisticsBySchema(SchemaName)

Input Parameter	Type	Description
SchemaName	string	Name of the schema.

Table 50 Returns of getActivityStatisticsBySchema(SchemaName)

Returns	Type	Description
ServiceName	string	Name of the service 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	The number of objects that were successfully identified for this schema which will be published or written to a file.
Error	string	The number of objects that were identified for this schema but were not published because the header of the schema failed validation for the Publication service, or was written to a file because the schema was not associated with the subscriber for a Subscription service.

getActivityStatisticsByService(ServiceName)

Returns statistics about the data handled by a given adapter service or all adapter services since the time the adapter was started.

Table 51 Input Parameter of getActivityStatisticsByService(ServiceName)

Input parameter	Type	Description
ServiceName	string	Name of service to get the statistics for. If no service name is given, performance statistics for all services is returned.

Table 52 Returns of getActivityStatisticsByService(ServiceName)

Returns	Type	Description
ServiceName	string	Service name.
SchemaName	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.

getAdapterServiceInformation()

Returns information about the services implemented by this adapter. The information is a summary of available adapter services.

Table 53 Input Parameter of getAdapterServiceInformation()

Input Parameter	Type	Description
serviceName	string	Name of the service from which to get information. Default is ALL.

Table 54 Returns of getAdapterServiceInformation()

Returns	Type	Description
Line	Integer	Sequential row number.
Name	string	Name of the Service.
Endpoint	string	Name of the endpoint used for this service.
Type	string	Type of the endpoint, for example, Publisher, Subscriber.
Quality of Service	string	Quality of service for the endpoint, for example, RV, RVCN.
Adapter Name	string	Name of the application for this sink.
Session Name	string	Name of the TIBCO Rendezvous session.
Subject	string	Subject defined for this 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.

Table 55 Input Parameters of `getComponents()`

Input Parameters	Type	Description
Component Name	string	Name of the component. Default is all components.
Component Type	string	Any of Publisher, Subscriber, Timer, or IODescriptor. The default value is All.

Table 56 Returns of `getComponents()`

Returns	Type	Description
Component Name	string	Name of the TIBCO Hawk component.
Instance ID	string	Name of this adapter instance.
Adapter Name	string	Name of the adapter.
Session Name	string	Name of the TIBCO Rendezvous session.
Component Type	string	The name of the TIBCO Adapter SDK class for this TIBCO Hawk component, such as MPublisher, MSubscriber, or MIODescriptorSource. For more information, see your TIBCO Adapter SDK documentation.
Description	string	Information about this TIBCO Hawk component, for example, time interval, signal type, validating publisher (or subscriber) and so on.

getConfig()

Retrieves generic configuration information. More specific configuration information is accessed through separate methods.

Table 57 getConfig()

Returns	Type	Description
Instance ID	string	Instance ID of this adapter.
Adapter Name	string	Name of the adapter.
Repository Connection	string	URL of the repository used for adapter configuration.
Configuration URL	string	Location of the adapter repository instance; 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.

Table 58 Input Parameter of getConfigProperties()

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.

Table 59 Returns of getConfigProperties()

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 repository instance file.

getConnectionStatistics()

Returns the state and statistics for all the current connections used by the adapter.

Table 60 *getConnectionStatistics()*

Returns	Type	Description
Connection ID	string	Unique identification of a particular connection.
Connection Type	string	Type or key that will match this connection to a thread or queue.
State	string	Current state: CONNECTED or DISCONNECTED.
NumRetries	integer	Total number of times this connection had to be reestablished.
TotalNumOperations	integer	Total number of operations processed by this connection since the adapter started.
CurrentNumOperations	integer	Total number of operations processed by this connection since the last reconnection.
NumLostConnections	integer	Total amount of time that this connection has been lost.
MeasurementInterval	integer	Displays the time (in seconds) since last time the adapter was reset, or if never reset, since the adapter started.

getHostInformation()

Returns standard and extended application information set.

Table 61 *getHostInformation()*

Returns	Type	Description
Name	string	Name of the property.
Value	string	Value of the property.

getPerfMonSetting()

Returns the setting of the perfMon option.

Table 62 *getPerfMonSetting()*

Returns	Type	Description
Setting	string	Value of the perfMon option.

getPollingBatchSize()

Returns the current polling batch size setting.

Table 63 Input Parameter of getPollingBatchSize()

Input Parameter	Type	Description
ServiceName	string	Name of the service to query.

Table 64 Returns of getPollingBatchSize()

Returns	Type	Description
PollingBatchSize	integer	Polling batch size in milliseconds.

getPollingInterval()

Returns the current polling interval setting.

Table 65 *getPollingInterval()*

Returns	Type	Description
PollingInterval	integer	Polling interval in milliseconds.

getQueueStatistics()

Returns 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.

Table 66 *getQueueStatistics()*

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.

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.

Table 67 Input Parameter of getRvConfig()

Input Parameter	Type	Description
Session Name	string	Name of the TIBCO Rendezvous session for which configuration is required (default is all).

Table 68 Returns of getRvConfig()

Returns	Type	Description
Instance ID	string	The instance 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. Additional methods are provided in [getConfig\(\)](#) on page 153 and [getRvConfig\(\)](#) on page 161.

Table 69 *getStatus()*

Returns	Type	Description
Instance ID	string	Instance 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.

getThreadStatistics()

Returns the operation counts of the current threads.

Table 70 *getThreadStatistics()*

Returns	Type	Description
ThreadID	string	Unique identification of a particular thread.
ThreadType	string	Type that tells what part of the adapter this thread belongs to. 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.

getTraceSinks()

Returns information about sinks to which traces currently go.

Table 71 Input Parameters of *getTraceSinks()*

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.

Table 72 Returns of *getTraceSinks()*

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”.

getVersion()

Retrieves version information for the current application. Two lines may be returned, one for the TIBCO Adapter SDK, one for the adapter.

Table 73 *getVersion()*

Returns	Type	Description
Instance ID	string	The instance ID as a string, for example, SDK.
Adapter Name	string	Name of the adapter as a string, for example, agentone.
Version	string	Version number as a string, for example, 1.1.

_onUnsolicitedMsg()

Displays all alert messages sent from the adapter or an error if not successful.

preRegisterListener()

Preregisters an anticipated listener.

Some sending applications can anticipate requests for certified delivery even before the listening applications start running. In such situations, the sender can preregister listeners, so TIBCO Rendezvous software begins storing outbound messages in the sender’s ledger. If the listening correspondent requires old messages, it receives the backlogged messages when it requests certified delivery.

Table 74 preRegisterListener()

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 listener to preregister.

Returns OK if the subscription service was preregistered successfully, false otherwise.

resetActivityStatistics()

Resets all the counts for the activity statistics.

resetConnectionStatistics()

Resets all the counts for the connection statistics.

resetThreadStatistics()

Resets all the counts for the thread statistics.

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.

Table 75 Input Parameters of reviewLedger()

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.

Table 76 Returns of reviewLedger()

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.
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 the ledger file.

setPollingBatchSize()

Sets the polling batch size for a service.

Table 77 *setPollingBatchSize()*

Input Parameter	Type	Description
ServiceName	string	Name of service where the polling batch size is set.
PollingBatchSize	integer	Polling batch size in milliseconds.

setPollingInterval()

Sets the polling interval for the Publication service.

Table 78 *setPollingInterval()*

Input Parameter	Type	Description
PollingInterval	integer	Polling interval in milliseconds.
ServiceName	string	Name of service where the polling interval is set.

setTraceSinks()

Adds a role or changes the file limit of a previously specified sink.

Table 79 *setTraceSinks()*

Input Parameters	Type	Description
Sink Name	string	Name of the sink for which to add a role or change the file limit.
Role Name	string	Name of the role 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()

Unregisters a currently preregistered listener.

Table 80 *unRegisterListener()*

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 listener to unregister.

This method returns `true` if the listener was unregistered successfully, `false` otherwise.

Appendix A **TIBCO Staging Tables**

TIBCO Staging tables are used to store the inserted and updated data in the Amdocs CRM system. The Publication service periodically polls these tables and publishes the new messages.

The following are the private TIBCO Staging tables that contain the outgoing messages and message numbers.

- `table_tibco_message_queue`
- `table_tibco_message_fields`

Topics

- [table_tibco_message_queue, page 178](#)
- [table_tibco_message_fields, page 179](#)

table_tibco_message_queue

table_tibco_message_queue contains the names of objects in the message queue.

Table 81 table_tibco_message_queue

Column Name	Description
message_num	Used to group the rows that form a message as a logic record.
Operation	The outgoing event name, for example createContact OR updateContact.
object_name	A valid Amdocs object name, for example, site or contact.
object_ref_name	<p>Any string that uniquely identifies a row within a message scope. This field is required because several rows of a message may have the same object_name.</p> <p>For example, a site update message may have several address rows, one for the shipment address, one for the billing address, and one for the primary address. In such a case, object_ref_name can be used to distinguish the different addresses.</p>
flag_existing_object	Currently not in use.
status	A long type field to indicate the status of a "logic" record.
status 0(UNPUBLISHED)	Represents messages yet to be published.
status 1(POLLED)	The message status is updated to 1 after the message has been retrieved for publishing.
status 2(PUBLISHED)	Represents a successfully published record.
status 3(PUBLISH_ERROR)	Represents that an error occurred and that the message was not published.
x_pub_id	A variable string type which holds the subject name.

table_tibco_message_fields

table_tibco_message_fields contains field values of the objects within a message.

Table 82 table_tibco_message_fields

Column Name	Description
message_num	Combined with object_ref_name to identify the fields of a row in table_tibco_message_queue.
obj_ref_name	Combined with message_num to identify the fields of a row in table_tibco_message_queue.
field_name	Valid Amdocs field name.
field_value	The value of this field.
field_type	An integer indicating if this field represents an object field (field_type = 1) or an object relation (field_type = 2).
field_data_type	<p>The Amdocs database type defined by Amdocs metadata table adp_sch_info.</p> <p>512 = long type; 514 = floating type; 516 = string type; 609= date-time type; 766 = decimal type.</p> <p>In an outgoing message, data-time type is published as a string; Amdocs floating type is published as a real number with 7 significant digits; a Amdocs decimal type is published as a real number with 15 digits of precision.</p>

Appendix B **Error Messages**

This appendix explains the error messages that are logged to the location specified when the adapter was configured.

Topics

- [Error Message Listing, page 184](#)

Error Message Listing

This table lists the various messages.

Table 83 Error Message (Sheet 1 of 39)

Message	Role	Category	Resolution
AEDTA-0000001	Waiting for Request...		
	Information	Adapter	Indicates normal adapter operation. No action required.
AEDTA-0000002	Exception in creating the Connection object:%1		
	Error	Adapter	Please see the exception message.
AEDTA-0000003	Invoked ConnectClfy		
	Information	Adapter	Indicates normal adapter operation. No action required.
AEDTA-0000004	Successfully connected to Amdocs		
	Information	Adapter	Indicates normal adapter operation. No action required.
AEDTA-0000005	Could not connect to Amdocs Database		
	Information	Adapter	Indicates normal adapter operation. No action required.
AEDTA-0000007	CBO Exception in getting the list of CBOs:%1'		
	Error	Adapter	Check if the database is up. Look for the error code in Amdocs documentation.
AEDTA-0000008	Query executed in getting the typeid :%1		
	Information	Adapter	Indicates normal adapter operation. No action required.
AEDTA-0000009	Query executed in getting the field names :%1		
	Information	Adapter	Indicates normal adapter operation. No action required.

Table 83 Error Message (Sheet 2 of 39)

Message	Role	Category	Resolution
AEDTA-0000010	CBO Exception in getting the field names and type :%1		
	Error	Adapter	Check if the database is up. Look for the error code in Amdocs documentation.
AEDTA-0000011	replied...		
	Information	Adapter	Indicates normal adapter operation. No action required.
AEDTA-0000012	CBO Exception in creating the application object for database connection :%1		
	Error	Adapter	Check if the database is up. Look for the error code in Amdocs documentation.
AEDTA-0000013	Creating session....		
	Information	Adapter	Indicates normal adapter operation. No action required.
AEDTA-0000014	Logging into the database....		
	Information	Adapter	Indicates normal adapter operation. No action required.
AEDTA-0000015	Successfully logged in....		
	Information	Adapter	Indicates normal adapter operation. No action required.
AEDTA-0000016	CBO Error in creating session and logging into the database		
	Error	Adapter	Check if the database is up. Look for the error code in Amdocs documentation.
AEDTA-0000017	Timer is not defined in the repository		
	Information	Adapter	Check whether the timer exists in the repository.

Table 83 Error Message (Sheet 3 of 39)

Message	Role	Category	Resolution
AEDTA-0000018	Exception in activating the Subscriber :%1		
	Error	Adapter	Check if the CLFYDTADiscSub subscriber is present in the repository.
AEDTA-0000019	Sending discovery message to find other running instances of CLFYDTA on subject %1 , service %2		
	Information	Adapter	Indicates normal adapter operation. No action required.
AEDTA-0000020	Exception in sending the discovery message :%1		
	Error	Adapter	Check if CLFY_DTARpcServer server is present in the repository.
AEDTA-0000021	Assuming master status.		
	Information	Adapter	Indicates normal adapter operation. No action required.
AEDTA-0000022	Received Reply to Discovery Request		
	Information	Adapter	Indicates normal adapter operation. No action required.
AEDTA-0000023	%1		
	Information	Adapter	Indicates normal adapter operation. No action required.
AEDTA-0000024	An instance of CLFYDTA is already available on : %1		
	Information	Adapter	Indicates normal adapter operation. No action required.
AEDTA-0000028	Exception in getting the address of the machine :%1		
	Error	Adapter	Please see the error message.
AEDTA-0000029	Exception in sending the reply :%1		
	Error	Adapter	Please see the error message.

Table 83 Error Message (Sheet 4 of 39)

Message	Role	Category	Resolution
AEDTA-0000030	Creating Application Object		
	Information	Adapter	Indicates normal adapter operation. No action required.
AEDTA-0000031	Logged out from the database		
	Information	Adapter	Indicates normal adapter operation. No action required.
AEDTA-0000032	Exited the Amdocs application		
	Information	Adapter	Indicates normal adapter operation. No action required.
AE-100001	Palette error/Adapter Configuration names must have only alphanumeric characters with no embedded spaces and can be up to 80 characters long.		
	Error	CONFIG	Please type in a valid name. Adapter Configuration names can only have alphanumeric characters and can be up to 80 characters long.
AE-100002	Palette error/The [%1] must be greater than [%2], and less than [%3].		
	Error	CONFIG	The value specified must be greater than or equal to 0, and less than or equal to 65535.
AE-100003	Palette Error/Names of adapters of the same type must be unique.		
	Warning	CONFIG	Specify a unique name for the adapter instance.
AE-100004	Connection Retry Mechanism Warning/This adapter version does not suspend services on connection failure. The configured value Number of Reconnect Attempts Before Suspending Impacted Service(s) will be ignored.		
	Warning	CONFIG	Indicates normal adapter operation. No action required.

Table 83 Error Message (Sheet 5 of 39)

Message	Role	Category	Resolution
AE-100006	JMS Service Configured/This adapter version %1 does not support JMS services, but one was found.		
	Warning	CONFIG	You have configured a service with transport type as JMS. But the AE Version of this adapter instance does not support JMS. You may change the AE Version or make sure that the runtime version is higher than this version.
AE-100007	XML Wire Format found/This adapter version %1 does not support XML Wire Format		
	Warning	CONFIG	You have configured a service with XML Wire Format. But the AE Version of this adapter instance does not support XML format. You may change the AE Version or make sure that the runtime version is higher than this version.
AE-100008	Mandatory Field Missing/The field [%1] is a mandatory field.		
	Error	CONFIG	Please specify a value for the field.
AE-100009	Palette error/The field [%1] must have only numeric values.		
	Error	CONFIG	Specify a valid numeric value for the field.
AE-100011	Invalid Service name:/Service name can only contain alphanumeric characters with no embedded spaces and cannot exceed maximum length of 80 characters.		
	Error	CONFIG	Please type in a valid name. Adapter Configuration names can only have alphanumeric characters and can be up to 80 characters long.
AE-110001	DTA Connection Failure/The Design Time Connection has failed %1.		
	Error	DTA	Verify if the Design Time Adapter is up and running.

Table 83 Error Message (Sheet 6 of 39)

Message	Role	Category	Resolution
AE-110002	Class Not Found/Could not find the Class. Error: %1. Error during connection to DTA.		
	Error	DTA	Please verify the palette jar which comes along with the is picked up by the adapter.
AE-110003	DTA Connection Successful/The Design-Time Adapter Connection is successful.		
	Information	DTA	Indicates normal adapter operation. No action necessary.
AE-110005	No DTA Connection/Connection to Design-Time Adapter is not established.		
	Warning	DTA	Check the connection parameters.
AE-120003	Palette Error/Get Schema Failed.%1.		
	Error	SRVC	Please check whether the Amdocs database is up.
AE-120012	DTA Connection Successful/CBO Connection parameters are valid. CBO test connection is successful.		
	Information	DTA	Indicates normal adapter operation. No action necessary.
AE-120013	DTA Connection Successful/JDBC Connection parameters are valid. JDBC test connection is successful.		
	Information	DTA	Indicates normal adapter operation. No action necessary.
AE-120014	DTA Connection Failed/Please check whether the database is up and Connection parameters are valid.		
	Information	DTA	Indicates normal adapter operation. No action necessary.

Table 83 Error Message (Sheet 7 of 39)

Message	Role	Category	Resolution
AE-120014	DTA Connection Failed/Please check whether the database is up and Connection parameters are valid.		
	Information	DTA	<p>If the database is up and Connection parameters are valid, check whether the following parameters in the <code>clarify.env</code> file have been added:</p> <ul style="list-style-type: none">• <code>jdbc_db_port=port</code>• <code>jdbc_db_name=database name</code>• <code>jdbc_db_server=database server</code> <p>These parameters are used by CBO.</p>
AE-120015	DTA CBO Connection Failed/CBO Connection parameters are invalid. Please check the parameters entered.		
	Information	DTA	Indicates normal adapter operation. No action necessary.
AE-120016	DTA JDBC Connection Failed/JDBC Connection parameters are invalid. Please check the parameters entered.		
	Information	DTA	Indicates normal adapter operation. No action necessary.
AE-120017	Do you want to Reload Schema?/Schema already Exists. Do you want to Reload Schema?		
	Warning	SRVC	Select the required option.
AE-120018	Palette error/Please enter the value in the specified format		
	Error	CONFIG	Enter the value in the format specified in the UI.
AE-120019	No CBOs found/There are no tables in the database starting with the given string.		
	Information	SRVC	Indicates normal adapter operation. No action necessary.

Table 83 Error Message (Sheet 8 of 39)

Message	Role	Category	Resolution
AE-120020	CBO not found/The selected CBO is not found in the database		
	Information	SRVC	Indicates normal adapter operation. No action necessary.
AE-100016	Get Schema failed/The schema was not fetched for the service. Please ensure that the aeschema file is writable.		
	Error	CONFIG	Please make sure that the aeschema file created for the service is writable.
AE-100017	Cannot Delete/The resource could not be deleted. Please ensure that the aeschema file is writable.		
	Error	CONFIG	Please make sure that the aeschema file created for the service is writable.
AE-100018	Cannot Paste Service/The service could not be pasted. Please ensure that the aeschema file is writable.		
	Error	CONFIG	Please make sure that the aeschema file created for the service is writable.
AE-100019	Cannot Rename/The adapter configuration could not be renamed. Please ensure that the aeschema file is writable.		
	Error	CONFIG	Please make sure that the aeschema file created for the service is writable.
AE-0000001	SDK Exception in Main : %1		
	Error	Adapter	Ensure all the required environment variables are set properly.
AE-0000002	Exception in Main : %1		
	Error	Adapter	Ensure all the required environment variables are set properly.
AE-0000003	Advisory Error Message : %1		
	Error	Adapter	Check configuration details for advisory.

Table 83 Error Message (Sheet 9 of 39)

Message	Role	Category	Resolution
AE-0000004	Advisory Info Message : %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000005	Advisory Warn Message : %1		
	Warning	Adapter	The details are displayed in the warning message.
AE-1000005	Adapter Successfully Initialized		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000006	Exception while processing the advisory event : %1		
	Error	Adapter	Ensure all the required environment variables are set properly.
AE-0000008	MException in Initializing the Adapter : %1		
	Error	Adapter	Ensure whether the required environment variables are set properly. see the <i>TIBCO ActiveMatrix Adapter for Amdocs CRM</i> for the required environment settings.
AE-0000009	Exception in Initializing the Adapter : %1		
	Error	Adapter	Ensure whether the required environment variables are set properly. see the <i>TIBCO ActiveMatrix Adapter for Amdocs CRM</i> for the required environment settings.
AE-0000010	The Adapter is shutting down		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000011	Connecting to Amdocs with the parameters : %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.

Table 83 Error Message (Sheet 10 of 39)

Message	Role	Category	Resolution
AE-0000012	Exception in Connecting to Amdocs during Initialization : %1		
	Error	Adapter	Please check for valid database connection parameters.
AE-910007	Unable to create connection with the Amdocs application using connection parameters [login=%1,password=****,DBServer=%2,DBName=%3] Amdocs Application error is : %4		
	Error	Adapter	Please check for valid database connection parameters.
AE-0000013	Got the Subscriber : %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000014	Activated the Subscriber (Name:Subject) = %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000015	No subscribers are configured in the Repository		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000016	Total Number of subscribers Activated : %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000017	No components in Repository to activate		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000018	Exception in Initializing Subscriber Components : %1		
	Error	Adapter	Check the Repository for proper configuration.

Table 83 Error Message (Sheet 11 of 39)

Message	Role	Category	Resolution
AE-0000019	Exception in Initializing Timer Components : %1		
	Error	Adapter	Check the Repository for proper configuration.
AE-0000020	Got the Publisher (Name:Subject) = %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000021	No Publishers are configured in the Repository		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000022	Total Number of Publishers Activated : %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000023	Failed in Initializing Publisher Components		
	Error	Adapter	Check configuration details for Publisher.
AE-0000024	Failed in Initializing Publisher Components		
	Error	Adapter	Check configuration details for Publisher.
AE-0000026	Exception in Initializing Publisher Components : %1		
	Error	Adapter	Check configuration details for Publisher.
AE-0000027	Terminate Subscriber not found in repository so shutting down the adapter		
	Error	Adapter	Please check the repository if terminate subscriber is configured.
AE-0000028	Got RPC Server : %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.

Table 83 Error Message (Sheet 12 of 39)

Message	Role	Category	Resolution
AE-0000029	No RPC Server are configured in the Repository		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000030	Total Number of RPC Servers Activated : %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000031	Exception in Initializing RPC Server Components : %1		
	Error	Adapter	Check configuration details for RPC Server.
AE-0000032	Initializing Adapter Services Info and Advisory Listeners		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000033	Adapter Services Info and Advisory Listeners initialized		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000034	Exception in Initializing Adapter Services Info and Advisory Listeners : %1		
	Error	Adapter	Check configuration details.
AE-0000035	Exception in initializing the user defined classes for user exits : %1		
	Error	Adapter	Check if the User Exits are properly configured in the repository and the user exit classes are there in the classpath.
AE-0000036	Registering Publisher Event %1 for Preprocessing		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000037	Registering Subscriber Event %1 for Preprocessing		
	Information	Adapter	Indicates normal adapter operation. No action necessary.

Table 83 Error Message (Sheet 13 of 39)

Message	Role	Category	Resolution
AE-0000038	Registering Subscriber Event %1 for Post processing		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000039	No User Exits defined for Publisher Preprocessing		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000040	No User Exits defined for Subscriber Preprocessing		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000041	No User Exits defined for Subscriber Post processing		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000042	Class Not Found Exception in Registering User Exits : %1		
	Error	Adapter	Please check if User Exit Class is included in the Classpath.
AE-0000043	Instantiation Exception in Registering User Exits : %1		
	Error	Adapter	Please check if User Exit Class is written as per directions in the <i>TIBCO ActiveMatrix Adapter for Amdocs CRM</i> .
AE-0000044	Illegal Access Exception in Registering User Exits : %1		
	Error	Adapter	Please check if User Exit Class is written as per directions in the <i>TIBCO ActiveMatrix Adapter for Amdocs CRM</i> .
AE-0000045	Exception in getting the User Exit Configuration from Repository : %1		
	Error	Adapter	Please check if the association attribute associated with user exits is present in the Repository.

Table 83 Error Message (Sheet 14 of 39)

Message	Role	Category	Resolution
AE-0000046	Successfully Logged into Amdocs database		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000047	CBO Error occurred while Connecting to database : %1		
	Error	Adapter	Please check the Amdocs documentation for the CBO Errors, check for proper connection parameters like login, password, database name and database server name.
AE-0000048	Exception occurred while Connecting to database : %1		
	Error	Adapter	Check for proper connection parameters like login, password, database name and database server name.
AE-0000050	CBO Error occurred while Creating a FormContext Object : %1		
	Error	Adapter	Please check the Amdocs documentation for the CBO Errors.
AE-0000051	Exception occurred while Creating a FormContext Object : %1		
	Error	Adapter	Please check the Amdocs documentation for the CBO Errors.
AE-0000052	CBO Error occurred while Checking the Amdocs Connection : %1		
	Error	Adapter	Please check the Amdocs documentation for the CBO Errors.
AE-0000053	Exception occurred while Checking the Amdocs Connection : %1		
	Error	Adapter	Check database connectivity.
AE-0000054	Successfully Logged into Amdocs database during reconnect		
	Information	Adapter	Indicates normal adapter operation. No action necessary.

Table 83 Error Message (Sheet 15 of 39)

Message	Role	Category	Resolution
AE-0000055	CBO Error occurred while Reconnecting to database : %1		
	Error	Adapter	Please check the Amdocs documentation for the CBO Errors. Also check whether database is up and running.
AE-0000056	Exception occurred while Reconnecting to database : %1		
	Error	Adapter	Please check whether database is up and running.
AE-0000057	Exception occurred while Waiting for the Connection Retry Interval : %1		
	Error	Adapter	Check database connectivity.
AE-0000059	Exited the Amdocs application		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000061	Exception occurred in Polling the database : %1		
	Error	Adapter	Check database connectivity.
AE-0000062	Exception in getting Error Publisher : %1		
	Error	Adapter	Please check if the Adapter Configuration in Repository Contains Error Handling Publisher.
AE-0000063	Adapter shutdown in progress		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000064	Inside OnEvent of Subscriber		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000065	Exception while extracting data from the incoming message		
	Error	Adapter	Please check if the incoming message is proper as per the Adapter documentation.

Table 83 Error Message (Sheet 16 of 39)

Message	Role	Category	Resolution
AE-920002	Subscription error. Subscription service %1 failed to deserialize the event received on subject %2 and SDK exception thrown is %3. The Repository URL is %4 and the Configuration URL is %5.		
	Error	Adapter	Please check if the incoming message is proper as per the Adapter documentation.
AE-920003	Subscription error. Subscription service %1 listening on subject %2 received an inbound event with null data. The Repository URL is %3 and the Configuration URL is %4.		
	Error	Adapter	Please check if the incoming message is proper as per the Adapter documentation.
AE-920007	Subscription error. Subscription service %1 listening on subject %2 received an inbound event with null class description. The Repository URL is %3 and the Configuration URL is %4.		
	Error	Adapter	Please check if the incoming message is proper as per the Adapter documentation.
AE-0000066	Exception while getting the optype field in the incoming message		
	Error	Adapter	Check if incoming message has the optype field.
AE-0000067	Exception while getting the wrapper_class field in the incoming message		
	Error	Adapter	Check if incoming message has the wrapper_class field.
AE-0000068	Exception while getting the main CBO name from the incoming message		
	Error	Adapter	Check if incoming message has the schema field.
AE-0000069	Exception while getting the lookup field from the incoming message		
	Error	Adapter	Check if incoming message has the lookup field.

Table 83 Error Message (Sheet 17 of 39)

Message	Role	Category	Resolution
AE-0000070	Exception while building the filter from the lookup in the incoming message		
	Error	Adapter	Check if the values passed in the lookup field are correct as per adapter documentation.
AE-0000071	Exception while extracting the main and contained CBO details		
	Error	Adapter	Check if incoming message has the schema field and is populated with correct values.
AE-0000072	Main CBO is : %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000073	%1		
	Error	Adapter	Check if the business data in the incoming message is proper.
AE-920015	Subscription error. Subscription service %1 listening on subject %2 failed due to application invocation error %3. The target application specific commands and parameters are %4		
	Error	Adapter	Check if the business data in the incoming message is proper.
AE-920016	Subscription error. Subscription service %1 listening on subject %2 received error %3 in PostProcessing user exit. The user exit parameters are %4		
	Error	Adapter	Check if the business data in the incoming message is proper.
AE-0000074	Reconnection Attempts failed. So shutting down the adapter		
	Error	Adapter	Check if Application is up and running.
AE-920017	Subscription error. Subscription service %1 listening on %2 could not send reply for the application invocation. The Repouurl is %3 and the configuration url is %4		
	Error	Adapter	Check if Application is up and running.

Table 83 Error Message (Sheet 18 of 39)

Message	Role	Category	Resolution
AE-940001	Request Response error. Request Response service %1 listening on %2 received unexpected null data in incoming request. The Repository URL is %3 and the Configuration URL is %4		
	Error	Adapter	Check if Application is up and running.
AE-940002	Request Response error. Request Response service %1 failed to deserialize the received MServerRequest to MInstance: Received event on subject %2, event = %3, SDK exception = %4. The Repository URL is %5 and the Configuration URL is %6		
	Error	Adapter	Check if Application is up and running.
AE-940007	Request Response error. Error in incoming data for RPC service: %1 on subject: %2. Missing mandatory parameter %3 for RPC input class %4		
	Error	Adapter	Check if Application is up and running.
AE-940008	Request Response error. Connection error in invocation of RPC service:%1 on subject:%2. Connection parameters are [login name=%3, password = ****, DBName =%4]		
	Error	Adapter	Check if Application is up and running.
AE-940009	Request Response error. Request Response service %1 listening on subject %2 failed due to application invocation error %3. The inbound event is %4.		
	Error	Adapter	Check if Application is up and running.
AE-940010	Request Response error. Request Response service %1 listening on subject %2 failed to create Reply Object with error %1.		
	Error	Adapter	Check if Application is up and running.
AE-950001	Request Response Invocation error. Request Response Invocation service %1 with subject as %2 received event from target application %3. It failed while converting event to Request, as it could not get the class description for %4. Repository URL is %5 and the Configuration URL is %6.		
	Error	Adapter	Check if Application is up and running.

Table 83 Error Message (Sheet 19 of 39)

Message	Role	Category	Resolution
AE-950002	Request Response Invocation error. Request Response Invocation service %1 with subject %2 received error while requesting event over the wire. The Request Response invocation endpoint details are [operationName =%3, className = %4].		
	Error	Adapter	Check if Application is up and running.
AE-950003	Request Response Invocation error. Request Response Invocation service %1 with subject %2 received null data while requesting event over the wire. The Request Response invocation endpoint details are [operationName =%3, className = %4].		
	Error	Adapter	Check if Application is up and running.
AE-950004	Request Response Invocation error. Request Response Invocation service %1 with subject %2 received timeout error while requesting event over the wire. The Request Response invocation endpoint details are [operationName =%3 , className = %4].		
	Error	Adapter	Check if Application is up and running.
AE-950005	Request Response Invocation error. Request Response Invocation service %1 with subject %2 received error while processing while reply. The Request Response invocation endpoint details are [operationName =%3, className = %4].		
	Error	Adapter	Check if Application is up and running.
AE-920014	Subscription error. Subscription service %1 listening on subject %2 could not process the inbound event due to connection error %3 against the application with parameters %4 after %5 connection retries. The connection timeout is %6.		
	Error	Adapter	Check if Application is up and running.
AE-000075	MException Received by Subscriber : %1		
	Error	Adapter	The sending and receiving applications may be using different repositories with different definitions of the same class or the sending application does not conform to the wire format.

Table 83 Error Message (Sheet 20 of 39)

Message	Role	Category	Resolution
AE-920001	Subscription error. Subscription service %1 listening on %2 received an Exception event type. The Repository URL is %3 and the Configuration URL is %4 Error received = %5		
	Error	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.
AE-0000076	Exception while processing the Event received by Subscriber : %1		
	Error	Adapter	The sending and receiving applications may be using different repositories with different definitions of the same class. Or the sending application does not conform to the wire format.
AE-0000077	Event Processed. Waiting for Next Event		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000080	Contained CBO is : %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000081	Exception Occurred while sending back the Subscriber Success Reply		
	Error	Adapter	Confirm that the subscriber is created.
AE-0000082	Exception Occurred while sending back the Subscriber Failure Reply		
	Error	Adapter	Confirm that the subscriber is created.
AE-0000083	Exception Occurred while sending back the Reply		
	Error	Adapter	Confirm that the subscriber is created.
AE-0000088	Error in preprocessing : %1		
	Error	Adapter	Check if user exit code is behaving properly.

Table 83 Error Message (Sheet 21 of 39)

Message	Role	Category	Resolution
AE-0000091	Error in post processing : %1		
	Error	Adapter	Check if user exit code is behaving properly.
AE-0000092	Error in Creating the Utility Class AppServer : %1		
	Error	Adapter	The program may be out of java virtual memory or another internal jvm exception occurred.
AE-0000093	Error in resetting the FormContext : %1		
	Error	Adapter	Check the database connectivity.
AE-0000094	Error in getting a reference to the CBO : %1		
	Error	Adapter	Please check in the incoming message if it contains the main and the contained CBO properly.
AE-0000095	Create Business Object succeeded Create Id is : %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000096	Create Business Object Failed		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000097	Update Business Object succeeded Update Id is : %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000098	Update Business Object Failed		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000099	"Requested operation %1 not supported"		
	Information	Adapter	Indicates normal adapter operation. No action necessary.

Table 83 Error Message (Sheet 22 of 39)

Message	Role	Category	Resolution
AE-0000100	Error in getting a reference to the queue : %1		
	Error	Adapter	Please check whether the business object is associated with a Queue.
AE-0000101	Error in getting a reference to the Login : %1		
	Error	Adapter	Please check in the incoming message if it contains the main and the contained CBO properly.
AE-0000102	Error in getting a reference to the Main CBO : %1		
	Error	Adapter	Please check in the incoming message if it contains the Main CBO properly
AE-0000103	"Field %1'could not be set in the Reply Object for the queried contained BO'%2"		
	Warning	Adapter	Check whether the schema used in the project is in sync with the database being used. If not, reload the schema.
AE-0000104	Error in building the Reply Message		
	Error	Adapter	Make sure that the server has sent the message correctly, make sure the parameters or the operation is specified properly.
AE-0000105	Trying to connect to application with the following parameters: %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000106	Error in getting the terminate Subscriber		
	Error	Adapter	Check if terminate subscriber endpoint exists.
AE-0000108	"Field %1'could not be set in the Reply Object for the queried contained BO'%2"		
	Warning	Adapter	Check whether the schema used in the project is in sync with the database being used. If not, reload the schema.

Table 83 Error Message (Sheet 23 of 39)

Message	Role	Category	Resolution
AE-0000201	Exception occurred in RpcServer Constructor : %1		
	Error	Adapter	The program may be out of java virtual memory or another internal jvm exception occurred.
AE-0000202	Exception occurred in OnInvoke of RpcServer : %1		
	Error	Adapter	Check for the correctness of operation name in the incoming request.
AE-0000203	Exception occurred while sending reply due to OnInvoke Error in RpcServer: %1		
	Error	Adapter	Check the TIBCO Rendezvous Settings.
AE-0000204	The request does not contain all the required parameters to perform a rpc operation		
	Error	Adapter	Check whether the incoming request contains all the required parameters.
AE-0000205	Reconnection Attempts failed. Attempting to Shut down the adapter		
	Error	Adapter	Check if Application is up and running.
AE-0000206	Exception Occurred due to mismatched / missing output parameter. Exception : %1		
	Error	Adapter	Check whether the output parameter in the schema is configured correctly.
AE-0000207	Exception Occurred while sending the reply back to the calling application in RpcServer. Exception : %1		
	Error	Adapter	Check the adapter and TIBCO Rendezvous Settings.
AE-0000208	Exception Occurred while invoking RpcServer Query Operation. Exception : %1		
	Error	Adapter	Check if the method name is correct/ Ensure that the database contains proper data.

Table 83 Error Message (Sheet 24 of 39)

Message	Role	Category	Resolution
AE-0000209	Exception Occurred while invoking RpcServer Workflow Operation. Exception : %1		
	Error	Adapter	Check if the method name is correct/ Ensure that the database contains proper data.
AE-0000210	Exception Occurred while invoking RpcServer Customized Operation. Exception : %1		
	Error	Adapter	Check if the method name is correct/ Ensure that the database contains proper data.
AE-0000211	Exception occurred while parsing the incoming request to retrieve input parameters. Exception : %1		
	Error	Adapter	Ensure that the incoming request schema conforms to the configured schema.
AE-0000212	Incoming request schema does not match with the configured schema. Exception : %1		
	Error	Adapter	Ensure that the incoming request schema conforms to the configured schema.
AE-0000213	RpcServer Query Operation invoked.		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000214	RpcServer Workflow Operation invoked.		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000215	RpcServer Customized Operation invoked.		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000216	Invocation of RpcServer Operation is completed. Waiting for Next Operation.		
	Information	Adapter	Indicates normal adapter operation. No action necessary.

Table 83 Error Message (Sheet 25 of 39)

Message	Role	Category	Resolution
AE-0000306	Query Operation is not supported for a %1 Business Object		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000307	WorkFlow Operation cannot be performed on a %1 Business Object		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000308	Customized Operation is not supported for a %1 Business Object		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000309	ActEntry Object cannot be created as query for contact returned no rows		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000310	Adapter does not support creation of BusOrg CBO		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000311	Case Object cannot be created as query for contact returned no rows		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000312	Contact Object cannot be created as query for site returned no rows		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000313	Adapter does not support creation of DemandDetail CBO		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000314	Adapter does not support updating a DemandDetail CBO		
	Information	Adapter	Indicates normal adapter operation. No action necessary.

Table 83 Error Message (Sheet 26 of 39)

Message	Role	Category	Resolution
AE-0000315	DemandHeader Object cant be Created as the query for contact returned no rows		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000316	Adapter does not support creation of Dialogue CBO		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000317	Adapter does not support updating a Dialogue CBO		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000318	DocInst Object cannot be Created as the query for contact returned no rows		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000319	Part Object cannot be Created as the query for part_num returned no rows		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000320	ServicePart Object cannot be Created as the query for contact returned no rows		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000321	Site Object cannot be created as query for address returned no rows		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000322	SubCase Object cannot be created as query for case returned no rows		
	Information	Adapter	Indicates normal adapter operation. No action necessary.

Table 83 Error Message (Sheet 27 of 39)

Message	Role	Category	Resolution
AE-0000323	%1 operation on %2 Business Object failed since the specified User does not exist in the database		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000324	%1 operation on %2 Business Object failed since the specified Queue does not exist in the database		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000325	%1 Business Object accepted in the default WIPbin		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000326	%1 Business Object accepted in the specified WIPbin.		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000327	Assign operation on %1 Business Object failed since the logged in user does not own the %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000328	%1 Business Object assigned to specified User and default WIPbin		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000329	%1 Business Object assigned to specified User and specified WIPbin		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000330	Dispatch operation on %1 Business Object failed since the logged in user does not own the %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.

Table 83 Error Message (Sheet 28 of 39)

Message	Role	Category	Resolution
AE-0000331	%1 Business Object dispatched to specified Queue		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000332	Forward operation on %1 Business Object failed since the logged in user does not own the %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000333	%1 Business Object forwarded to specified Queue		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000334	Move operation on %1 Business Object failed since the logged in user does not own the %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000335	%1 Business Object moved to default WIPbin		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000336	%1 Business Object moved to specified WIPbin		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000337	Reject operation on %1 Business Object failed since the logged in user does not own the %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000338	%1 Business Object rejected		
	Information	Adapter	Indicates normal adapter operation. No action necessary.

Table 83 Error Message (Sheet 29 of 39)

Message	Role	Category	Resolution
AE-0000339	%1 Business Object Yanked		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000340	Exception occurred in the init method of %1 wrapper. Exception : %2		
	Error	Adapter	Check CBO related settings
AE-0000410	Published messages will be deleted from the database		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000411	Published messages will be retained in the database		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000414	Exception in getting the Publisher options from the Repository :%1		
	Error	Adapter	The association attribute Publisher options may have been deleted from the repository. Enter the values in the Publisher options tab and save the repository.
AE-0000416	Exception in polling the database :%1		
	Error	Adapter	Please check the exception message for details.
AE-0000420	SQLException in updating the publish status in the TIBCO_MESSAGE_QUEUE table :%1		
	Error	Adapter	Check if the database is up. Look for the error code in documentation.
AE-0000422	General Exception in clearing the queue and updating the status in the staging table : %1		
	Error	Adapter	Please check the exception message. Your system may be running short of memory. Please close some applications and restart the adapter.

Table 83 Error Message (Sheet 30 of 39)

Message	Role	Category	Resolution
AE-0000423	Exception in creating the string of published message numbers whose status is to be updated : %1		
	Error	Adapter	Please check the exception message. Your system may be running short of memory. Please close some applications and restart the adapter.
AE-0000424	Exception in updating the queue containing the published message numbers : %1		
	Error	Adapter	Please check the exception message. Your system may be running short of memory. Please close some applications and restart the adapter.
AE-0000427	Exception in getting message from the message queue :%1		
	Error	Adapter	Please check the exception message. Your system may be running short of memory. Please close some applications and restart the adapter.
AE-0000430	Exception in deleting message from the message queue :%1		
	Error	Adapter	Please check the exception message.
AE-0000431	SQL Exception in getting distinct message numbers from the queue table :%1		
	Error	Adapter	Check if the database is up. Look for the error code in documentation.
AE-0000432	General Exception in getting distinct message numbers from the queue table :%1		
	Error	Adapter	Please check the exception message.
AE-0000436	Exception in creating the message queue :%1		
	Error	Adapter	Please check the exception message.

Table 83 Error Message (Sheet 31 of 39)

Message	Role	Category	Resolution
AE-0000437	SQLException in querying of message fields from queue and field tables :%1		
	Error	Adapter	Check if the database is up. Look for the error code in documentation. If datatype found is CLOB turn "adclycrm.oracle.useCLOB" option in tra to ON.
AE-0000438	General Exception in bulk querying of messages with status 0 from queue and field tables :%1		
	Error	Adapter	Please check the exception message.
AE-0000440	Exception in initializing the publisher threads :%1		
	Error	Adapter	Please check the exception message.
AE-0000441	Interrupted thread exception :%1		
	Error	Adapter	Please check the exception message. Restart the adapter.
AE-0000445	Message to be published :%1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000447	MException in getting the Publisher specific schemas :%1		
	Error	Adapter	Please make sure the schemas created under the folder PubSchema in the repo are present.
AE-0000448	Exception in Processing the message to be published :%1		
	Error	Adapter	Please check the user-defined class written for implementing PrePublisher User exit.
AE-0000452	Exception in casting the field values :%1		
	Error	Adapter	Please make sure field_value in the TIBCO_MESSAGE_FIELDS table is of the same data type as specified in the field _data_type column.

Table 83 Error Message (Sheet 32 of 39)

Message	Role	Category	Resolution
AE-0000453	Exception in getting the Publisher specific sequences which are required to construct the message:%1		
	Error	Adapter	Please make sure the default sequences created under the SchemasSequences..... folder in the repo are present.
AE-0000458	Exception in invoking the onInvoke method of the object of UserDefined class :%1		
	Error	Adapter	Please check the user-defined class written for implementing PrePublisher User exit.
AE-0000461	Exception in querying queue/field table or in creating the MInstance due to incorrect data in field table : %1		
	Error	Adapter	See the exception message for details.
AE-0000462	Exception in freeing database connection on SQLException in getting distinct message numbers : %1		
	Error	Adapter	Check if the database is up. Look for the error code in documentation.
AE-0000463	Server name is not provided in the TRA file		
	Error	Adapter	Please provide the server name.
AE-0000464	Port number is not provided in the TRA file		
	Error	Adapter	Please provide the Port number.
AE-0000465	No CBO connection is made with the database as no subscriber and rpc are configured		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000467	Trying to connect using jdbc with the following parameters : %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.

Table 83 Error Message (Sheet 33 of 39)

Message	Role	Category	Resolution
AE-0000468	Exception in registering the driver :%1		
	Error	Adapter	Check if the classes12.zip is present in the classpath.
AE-0000469	SQLException in obtaining connection cache :%1		
	Error	Adapter	Check the connection parameters.
AE-0000470	Exception in obtaining connection cache :%1		
	Error	Adapter	Check the connection parameters.
AE-0000471	Exception in connecting to the database using JDBC : %1		
	Error	Adapter	Check the connection parameters.
AE-0000472	Duplicate subject names have been configured for RPC Clients. The adapter will shut down.		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000473	Error in creating run time subscribers for RPC client :%1		
	Error	Adapter	Check the RvSession name given in the palette already exists.
AE-0000474	RpcClient : %1 associated with Subscriber listening on Subject :%2 and Session :%3 found		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000475	ClientRequest for RPC Client Endpoint :%1 created		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000476	Type of client operation : %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.

Table 83 Error Message (Sheet 34 of 39)

Message	Role	Category	Resolution
AE-0000477	Error in creating ClientRequest object :%1		
	Error	Adapter	Please check the ClassName and OperationName configured for the endpoint.
AE-0000478	Error in creating MInstance for ClientSchema :%1		
	Error	Adapter	Check the configuration details.
AE-0000479	Request class associated with the clientRequest schema : %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000480	Error in creating MInstance of requestSchema class :%1		
	Error	Adapter	Check the configuration in the repository.
AE-0000482	Error in parsing the incoming TIBCO Rendezvous message and setting the values for the request schema: %1		
	Error	Adapter	Check the incoming TIBCO Rendezvous message is of the proper format.
AE-0000484	Error in setting the values for the MClientRequest parameters : %1		
	Error	Adapter	Check the configuration in the repository.
AE-0000485	Reply from the external server has exception		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000487	Request timed out before the reply arrived : %1		
	Error	Adapter	Check whether the external RPC server is up and running.
AE-0000488	Error due to missing class description or failure to marshall/unmarshall data : %1		
	Error	Adapter	Metadata class information of the repository and the MB should be the same.

Table 83 Error Message (Sheet 35 of 39)

Message	Role	Category	Resolution
AE-0000489	Error in parsing the server reply and creating the reply TIBCO Rendezvous message : %1		
	Error	Adapter	Check whether the reply is in the required format.
AE-0000490	Error :%1		
	Error	Adapter	Check the incoming TIBCO Rendezvous message is of the proper format.
AE-0000491	Error in creating the errorReply message : %1		
	Error	Adapter	Check whether the proper reply schema is configured.
AE-0000492	RvReply message : %1		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000493	RPC client operation invoked successfully		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000494	Error in invoking the client operation :%1		
	Error	Adapter	Check the configuration in the repository
AE-0000495	No CBO connection is made with the database		
	Information	Adapter	Indicates normal adapter operation. No Subscription or Request-Response Services are configured.
AE-0000496	Main CBO is not set in the schema. Message will not be published.		
	Error	Adapter	Use the Schema tab in the Publication service to set the Main CBO.

Table 83 Error Message (Sheet 36 of 39)

Message	Role	Category	Resolution
AE-0000497	Contained CBO %1 is not found in the schema		
	Warning	Adapter	Please see the <i>TIBCO ActiveMatrix Adapter for Amdocs CRM</i> for the steps to be followed to attach a contained CBO.
AE-0000498	Schema mismatch problem : %1		
	Warning	Adapter	Check whether the schema used in the project is in sync with the database being used. If not, reload the schema.
AE-0000499	Error in trying to remove an expired connection : %1		
	Error	Adapter	Check if the DB is up.
AE-0000500	Error in invalidate connection : %1		
	Error	Adapter	Check the validity of DB connection parameters.
AE-0000501	Error while clean up of connection pool : %1		
	Error	Adapter	Check if the DB is up.
AE-0000502	Error in removing connection event listener : %1		
	Error	Adapter	Check the validity of DB connection parameters.
AE-0000503	Not able to connect to database. Please check the database parameters in the tra and repository		
	Information	Adapter	Check the database parameters in the tra and repository.
AE-0000504	Exception in initializing the RPC Client components : %1		
	Error	Adapter	Please check the repository for proper configuration.

Table 83 Error Message (Sheet 37 of 39)

Message	Role	Category	Resolution
AE-0000505	%1 field is not found in the configured request schema		
	Warning	Adapter	Check whether the schema used in the project is in sync with the database being used. If not, reload the schema.
AE-0000507	Startup Warning. Password decryption was unsuccessful.		
	Warning	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000508	Using CLOB data type instead of LONGVARCHAR		
	Warning	Adapter	Indicates normal adapter operation. It also indicates that Oracle 9i or Oracle 10g database is being used. No action necessary.
AE-0000511	Exception occurred while getting connection from the connection pool.		
	Error	Adapter	Check for connection pool size. The size should not be less than zero.
AE-0000512	Exception occurred in associating connection to a thread.		
	Error	Adapter	Please check if the specified number of threads and connections have been created.
AE-0000515	%1 Connections created with Application.		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000517	Stopped the dispatchers.		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000519	%1 dispatchers created.		
	Information	Adapter	Indicates normal adapter operation. No action necessary.

Table 83 Error Message (Sheet 38 of 39)

Message	Role	Category	Resolution
AE-0000524	Exception occurred in associating connection to a thread.		
	Warning	Adapter	Please check if the specified number of threads and connections have been created.
AE-0000526	Exception in adding threadId to table":%1.		
	Warning	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000527	Reconnect Attempt %1 succeeded.		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000528	Suspending Subscribers since reconnection failed in transient attempts..		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-0000529	Activating Subscribers after reconnection success.		
	Information	Adapter	Indicates normal adapter operation. No action necessary.
AE-930001	Publication error. Publication service %1%2 encountered error %3 while trying to connect to target application %4. Connection parameters are %5, the connection timeout is %6 milliseconds, and the number of retry efforts is %7.		
	Error	Adapter	Check the target application and make sure it is up and running. Check the connection parameters for right syntax and values.
AE-930003	Publication error. It failed while converting event to Minstance as it could not get the class description. Repository URL is %3 and the Configuration URL is %4.		
	Error	Adapter	Please verify the configuration of the Publication service and check that the schema/class definitions are present in the repository.

Table 83 Error Message (Sheet 39 of 39)

Message	Role	Category	Resolution
AE-930009	Publication error. Publication service %1 with publication subject %2 received an event from the target application but encountered error %3 in pre-processing user exit invocation. The target application details are %6.		
	Error	Adapter	Make sure that the parameters passed to UserExit are valid and the User Exit can be invoked by the adapter.
AE-930014	Publication error. Publication service %1 with publication subject %2 received error while sending event over the wire.%3		
	Error	Adapter	Please check repository settings for valid configuration of the publish endpoint for this service.



The error messages for Request-Response Invocation Service cannot exceed 255 characters when used in the flexible deployment scenario. This is a limitation of the Tuxedo middleware.

Appendix C **User Exits**

This chapter describes custom processing that can be performed for messages depending on the adapter component.

Topics

- [Overview, page 224](#)
- [How the Adapter Implements User Exits, page 225](#)

Overview

User exits are hooks provided by the adapter. They allow you to perform custom processing on each message depending on the adapter (Subscription or Publication service) and for the message, which is currently being processed. Custom processing can be done before (pre-processing) and after (post-processing) the original message is processed.

How the Adapter Implements User Exits

The adapter contains an interface implemented for the UserExits functionality. The supported functionalities are:

- Pre-processing of the Subscription service event message
- Post-processing of the Subscription service event message
- Pre-processing of the Publication service event message

To provide a user exit for a Subscription Service, or a Publication service event, you need to define a class that implements the IUserExit Interface. You have to compile the class and include it in the adapter's classpath.

Pre-processing examples include data validation, message format conversion. An example of post-processing would be generating additional trace messages, validation of the reply object and any format modifications.

To implement a user exit for the adapter component:

1. For the adapter components (Subscription service and Publication service), select the event(s) to provide the user exit. For example, TEST.CASE event.
2. Choose when you want the adapter to call the user exit: before or after the event is processed.
3. Define a new class, which implements the IUserExit Interface (example: presubevent.java for implementing pre-processing of Subscription Service event message).
4. Implement the methods defined in the interface (example: onInitialization, onInvoke, onTermination).

Implement User Exits for Pre-processing of Publication Service Event Message

1. In the adapter configuration, select the **Configuration** tab and select the **Show all Tabs** checkbox.
2. Select the **General** tab.
3. In the Pre-Publication, Pre-Subscription and Post-Subscription fields, enter the event name and class name separated by a semi colon in the respective fields. Example:
TEST.CASE:presubevent

Each pair of event and class should be separated by a semicolon. Example:
update_case:prepubevent;TEST.CASE:prepubevent

4. Click the **Apply** button to save the event and the class in the project.

IUserExit Interface

Given next is the IUserExit Interface.

```
public interface IUserExit
{
    void onInitialization(AdapterCore m_mApp);
    void onTermination();
    UserExitResult onInvoke(MInstance incomingEvent);
    void onInvoke(MInstance incomingEvent,boolean bSuccess,Exception Ex);
}
```

Example:

The following steps show how to create a user exit implementation for pre-processing, for the event TEST.CASE for the Subscription Service component.

1. Define a class that implements the IUserExit interface. Following is the definition for the example:

```
import com.tibco.adapter.*;
import com.tibco.sdk.metadata.*;

public class presubevent implements IUserExit{
    AdapterCore m_mApp;
    UserExitResult userExitRes;

    public void onInitialization(AdapterCore m_mApp){
        this.m_mApp = m_mApp;
        System.out.println("Inside OnInitialization in PubEvent");
    }

    public void onTermination(){
        System.out.println("Inside onTermination in PubEvent");
    }

    public UserExitResult onInvoke(MInstance incomingEvent){
        System.out.println("Inside onInvoke in PubEvent");
        userExitRes=new UserExitResult();
        userExitRes.setStatus(true);
        userExitRes.setInstance(incomingEvent);

        return userExitRes;
    }
}
```

2. Implementation for the methods onInitialization, onTermination, and onInvoke (onInvoke is the method that is called for every registered event) should be provided as above. OnInitialization and onTermination are called only once for every event exit implementation.

In the case of the Preprocessing of Publication service and Subscription service event message, onInvoke method takes an MInstance object and returns an object of class

UUserExitResult. UserExitResult class provides three functions: setStatus(boolean status), setInstance(MInstance incomingInstance), and setException(Exception).

In the case of Postprocessing of Subscription service event message, onInvoke method takes an MInstance object, boolean and Exception object as parameters and returns none.

3. Compile this class.

Before compiling, make sure that adclyCRM.jar which comes with adapter installation is in the classpath. The compiled class is included in the adapter classpath. This can be done by appending the location of the class in the tibco.env.CUSTOM_CP_EXT variable in the adclyCRM.tra file.

Appendix D **Wrapper Classes**

This chapter describes Wrapper classes, how to write wrapper classes and how they work with the adapter.

Topics

- [Overview, page 230](#)
- [Wrappers Packaged with the Adapter, page 231](#)
- [Wrapper Class and TibcoCBOInterface, page 237](#)
- [Writing Wrapper Class for Case Business Object, page 238](#)

Overview

A Wrapper class provides more functionality than its underlying business object. It encapsulates business logic which helps by extending the base CBOs, which enables creating or updating CBOs. All the wrapper classes should implement `TibcoCBOInterface`, which contains all the operations that can be performed. The usage of `TibcoCBOInterface` ensures that all the wrapper classes have a similar structure.

Objectives of Wrapper Classes

- Provide more functionality than its underlying CBO
- Encapsulate Business logic to create or update CBOs
- Enable customization to implement the required functionality

Wrappers Packaged with the Adapter

This section explains the wrapper classes that are packaged with the adapter installation and the functionality provided for each of the wrapper classes.

Wrappers are available in the following location:

TIBCO_HOME\adapter\adclycrm\version_num\samples\wrapper

ActEntry_Wrapper

This class is used as a wrapper over the `ActEntry` CBO and it extends the `ActEntry` class. It can be used to perform activities like update, queryRPC, workflowRPC operations on the CBO.

The `ActEntry_Wrapper` class is used to create new activity log entry for business objects (like installed part or workflow item). The wrapper is customized to associate new activity log entry with a Business Object (BO) such as Case, get activity log entries for Business Object, and associate a Business Object as a participant in an activity.

Address_Wrapper

This class is used as a wrapper over the `Generic` CBO and it extends the `Generic` class. It can be used to perform activities like Create, Update and QueryRPC operations on the CBO.

BusOrg_Wrapper

This class is used as a wrapper over the `BusOrg` CBO and it extends the `BusOrg` class. The `BusOrg` wrapper class is used to perform create, update and QueryRPC operations on the CBO that stores information about the organization that you do business with.

Case_Wrapper

This class is used as a wrapper over the `Case` CBO and it extends the `Case` class. The `Case` wrapper class is customized to perform create, update, QueryRPC, and WorkflowRPC operations on the CBO.

The wrapper class can fetch billto/shipto address, commitlog, contact, emaillog, noteslog, onsitelog, phonelog, researchlog, site and sitepart associated with a given case. The wrapper class also performs Accept, Assign, Dispatch, Forward, Move, Reject and Yank workflow operations for a given case.

Communication_Wrapper

This class is used as a wrapper over Communication CBO and it extends the Communication class. It can be used to perform Create, Update and QueryRPC operations on the CBO.

The Communication wrapper class can fetch and modify communication information from the Communication table (like address, com role or information related to email communication).

Contact_Wrapper

This class is used as a wrapper over the Contact CBO and it extends the Contact class. The Contact wrapper class is customized to perform create, update, QueryRPC, and WorkflowRPC operations on the CBO.

The Contact wrapper class can fetch and modify contact information from the contact table (like account information, action items, role, credit card information, email logs, alerts, opportunities, phone logs, Quote, sites associated with contact). The wrapper can also be used to associate or disassociate a contact with a site with a given specific role. Updates site information for a particular contact. Logs a phone call.

CreditCard_Wrapper

This class is used as a wrapper over the CreditCard CBO and it extends the CreditCard class. The CreditCard wrapper class is customized to perform create, update, QueryRPC, and WorkflowRPC operations on the CBO.

DemandDetail_Wrapper

This class is used as a wrapper over the DemandDetail CBO and it extends the DemandDetail class. The DemandDetail wrapper class is customized to perform QueryRPC, and WorkflowRPC operations on the CBO.

Limitation

Create and update operations are not supported by the wrapper class because of the limitation of the adapter to perform these operations. The DemandDetail CBO is basically intended for use with eOrder web application and provides basic functionality for fulfilling online orders. The CBO is basically used in conjunction with the demandHeader Business Object to define part requests for a customer order.

The wrapper class can be customized to only fetch contained objects which provide details like current state of part request. The wrapper class can be customized to update the existing content of the contained class only.

DemandHeader_Wrapper

This class is used as a wrapper over the DemandHeader CBO and it extends the DemandHeader class. The DemandHeader wrapper class is customized to perform Create, Update, QueryRPC, and WorkflowRPC operations on the CBO.

The wrapper class is also customized to create a new demand header, establish user's shipping and billing information, and generate part requests for the items in the user's order. The wrapper class can fetch information like shipping and billing information and the details of the person who requested the parts.

Limitation

The behavior of the DemandHeader business object has been tailored for use with eOrder. You should use it only to initiate order fulfillment. Do not attempt to use the business object to create other types of part requests in ClearLogistics.

Dialogue_Wrapper

This class is used as a wrapper over the Dialogue CBO and it extends the Dialogue class. The Dialogue wrapper class is customized to perform QueryRPC, and WorkflowRPC operations on the CBO.

The class can fetch the Communication Business Object associated with the Dialogue Business Object. The wrapper class also supports the workflow operations like Assign, Dispatch and Yank workflow operations for a given case.

Limitation

The wrapper class cannot be used to support create and update operations because of the limitation of the adapter to support these operations on the Dialogue CBO.

The assign operation can be performed only if the incoming request contains information about the user to whom the object has to be assigned.

Dispatch and Forward operations can be performed only if the incoming request contains information about a Queue to which the object has to be dispatched.

DocInst_Wrapper

This class is used as a wrapper over the DocInst CBO and it extends the DocInst class. The DocInst wrapper class is customized to perform create, update, QueryRPC, WorkflowRPC and CustomizedRPC operations on the CBO. The wrapper class can create new attachments, get attachment objects such as case, subcase, change request and site. The wrapper class fetches information like path of storage of the attachment, and so on.

Generic_Edr_Com_Role Wrapper

This class is used as a wrapper over Generic CBO and it extends the Generic class. It is customized to perform Create, Update and QueryRPC operations on the CBO.

This wrapper class can fetch the information about Address BO associated with a ComRole BO.

NotesLog_Wrapper

NotesLog wrapper is used as a wrapper over NoteLog CBO and it extends NotesLog class. It can be used to perform create, update and queryRPC operations on the CBO.

This can create a new notes log or fetch the existing noteslog for a case, contract or subcase.

OEQuote_Wrapper

This class is used as a wrapper over the OEQuote CBO and it extends the OEQuote Class. The OEQuote wrapper class is customized to perform create, update, QueryRPC, WorkflowRPC and CustomizedRPC operations on the CBO.

The wrapper class can be used to access the contract table. It can be customized to provide a list of products and options being purchased, price of items, price adjustments like discounts and surcharge, organization or individual purchase item, shipping and billing information for the organization or the user, payment information, phone logs, email logs and notes associated with the order.

The wrapper class can be customized to access information in OEQuote table in addition to the contract table. It is also customized to perform workflow operations like dispatch, assign, accept, and yank quotes and part requests.

Limitation

The Dispatch and Forward operations can be performed only if the incoming request contains information about a Queue to which the object has to be dispatched.

Assign operation can be performed only if the incoming request contains information about the user to whom the object has to be assigned.

PartPrice_Wrapper

This class is used as a wrapper over the PartPrice CBO and it extends the PartPrice class. The PartPrice wrapper class is customized to perform create, update, QueryRPC, WorkflowRPC and CustomizedRPC operations on the CBO.

The wrapper class is customized to get pricing information for a specific part. The wrapper class can be customized to access contained objects of the PricePart to list the parts, currency, and price book associated with price instance.

Limitation

Although you can use this business object to add new price instance information to the database, you must be very careful while doing so. This business object does not validate the price instance data you add. To make sure your price instance data is validated properly, use the Product Manager application to enter your data.

Part_Wrapper

This class is used as a wrapper over the Part CBO and it extends the Part Class. The Part wrapper class is customized to perform create, update, QueryRPC, WorkflowRPC and CustomizedRPC operations on the CBO. The class is also customized to get information from part_num table about the part number or child part.

ServicePart_Wrapper

This class is used as a wrapper over the ServicePart CBO and it extends the ServicePart Class. The ServicePart wrapper class is customized to perform create, update, QueryRPC, WorkflowRPC and CustomizedRPC operations on the CBO. The wrapper class can be customized to access the site_part table in the Amdocs CRM system. The wrapper can also get the related sites and related products for a service part.

ShopList_Wrapper

This class is used as a wrapper over the ShopList CBO and it extends the ShopList Class. The ShopList wrapper class is customized to perform create, update, QueryRPC, WorkflowRPC and CustomizedRPC operations on the CBO.

Site_Wrapper

This class is used as a wrapper over the Site CBO and it extends the Site Class. The Site wrapper class is customized to perform create, update, QueryRPC, WorkflowRPC and CustomizedRPC operations on the CBO.

The wrapper class is also customized to perform operations like create new site, get and modify site information in the site table of the Amdocs CRM system. The class can get and change the address and primary business organization associated with the site, install or link a part at a site and view existing log entries for a site.

SubCase_Wrapper

This class is used as a wrapper over the SubCase CBO and it extends the SubCase class. The SubCase wrapper class is customized to perform create, update, QueryRPC, WorkflowRPC and CustomizedRPC operations on the CBO. The SubCase wrapper class is customized to fetch address, closelog, commitlog, contact, emaillog, noteslog, onsitelog, phonelog, researchlog, site and sitepart associated with the SubCase. Also customized to perform workflow operations like Accept, Assign, Dispatch, Forward, Move, Reject, and Yank operations.

Task_Wrapper

This class represents the task table of the Generic Business Object. It extends Generic class, which contains information about other business objects and action items associated with business organizations. The Task_Wrapper class can be used to perform Create and QueryRPC operations on the CBO.

It has a **task_for2bus_org** relationship with Bus_Org CBO. The Bus_Org CBO is a contained CBO of the Task_Wrapper class.

User_Wrapper

This class is used as a wrapper over User CBO and it extends User class. It can be used to perform Create, Update and QueryRPC operations on the CBO.

It can also be used to fetch the details of current user's site,skills and also agent, employee, wipbin and actions of the current user.



The wrappers can be enhanced to do certain customized operations by implementing the customizedRPC method in the wrapper. The Case wrapper provides a reference custom operation implementation. The custom operation in the Case wrapper populates the tibco_ex_case_history table with the values from case table.

Wrapper Class and TibcoCBOInterface

Wrapper classes are developed to make the task of creating or updating a CBO easier.

The wrapper class extends an interface called `TibcoCBOInterface`. The usage of `TibcoCBOInterface` ensures that all the wrapper classes have a similar structure. The defined structure acts as a template, enabling you to write a new wrapper class.

The interface contains a list of methods, which a wrapper class must implement. These methods are:

1. `Init`—This method creates an object of the wrapper class and the handle is used throughout the session. For wrappers extending `Generic` class, after creating an object of the wrapper class and creating a handle to it, call `setDBObjectName()` present in `Generic` class to specify the table or view to which this CBO provides access. Please refer `Generic_Edr_Com_Role_Wrapper` class provided in the adapter samples for more details.
2. `Create`—This method, when implemented by a wrapper class, should contain the business logic to create a CBO. For create operation, call `GenericHandle2CBO` defined in `AppServer` class to get an object reference for the contained CBO. The details provided for Contained CBO in the input should be that of an existing entry in the database. For example, to create a new contact, the details of an existing site have to be provided in the input.
3. `Update`—This method, when implemented by a wrapper class, should contain the business logic to update a CBO.
4. `queryRPC`—This method should be implemented by all the wrapper classes in such a way that it should incorporate the logic to extract details from the Amdocs CRM system for a particular Business Object based on the user request.
5. `workflowRPC`—This method should be implemented by all the wrapper classes in such a way that it should incorporate the logic to perform workflow operations on workflow Business Objects.
6. `customizedRPC`—This method should be implemented by all the wrapper classes in such a way that it should incorporate the logic to perform customized operations on CBOs.

Writing Wrapper Class for Case Business Object

Steps Involved

Example:

1. The class implements `TibcoCBOInterface`.
2. The class extends the derived CBO `Case`.
`Public class Case_Wrapper implements TibcoCBOInterface extends Case`
3. Implement the method `Init()` of the Interface. The method creates an object of the wrapper class and creates a handle for it.
`Case_Wrapper boCase = (Case_Wrapper)this;`
4. Implement the method `create()` of the interface. When invoked, this method inserts a new row for a business object, along with the details in the corresponding table in the Amdocs CRM system. The input parameters for this method carries all the information required to perform the operation.

Algorithm:

- Get an object reference for the contained CBO. For `Case`, the contained Business Object is `Contact`. This can be done by calling a method `getHandle2CBO` defined in `AppServer` class. `AppServer` class contains all the common utility functions used in the adapter. The function `getHandle2CBO` takes two parameters, one holding the underlying table name of the business object and the other pointing to its Derived Business Object Name. For contact business object, the table name is `contact` and its business object name is `Contact`.
 - Invoke `addNew()` method to add a new row to the business object. This method assigns a temporary object ID to the new row. When the object is updated for the first time, a permanent object ID will be assigned to the row before committing the changes to the database.
 - Call `setCBOValues` method defined in the `AppServer` to fill the field values in the newly added row. The field values will be taken from the incoming request. If the incoming request does not contain a value for a field, its value is set to the default value (if there is any). Otherwise it is set to null.
 - To associate the newly created case with the contact, use `getContact().add()` passing the contact object reference.
 - Call `update()` method to commit the changes to the database.
5. Implement the `update()` method of the interface. When invoked, this method will update the information associated with a business object. The adapter provides functionality to change the field values associated with a business object. It does not support changing the reference associated with contained/child business objects.

- Algorithm:**
- Get a handle to the Case CBO to be updated. This can be done by calling `getMainCBO()` of `AppServer` class by passing the Case business object handle as a parameter.
 - Call `setCBOValues()` to update the field values.
 - Invoke `update()` to commit the changes to the database.
6. Implement the `queryRPC()` method of the interface. This method fetches information about the child business objects (or contained CBO(s)) based on the request and sends the result back to the calling function. The adapter provides support to fetch information about the contained CBOs and if you want to get data associated with field values of a CBO, you have to customize the wrapper class. The incoming request should contain the name of the method to be invoked to perform the query operation. For example, if you want to get information about the Site associated with a Case, the method name to be used is `getsite`. The method name should reflect the name of the method defined in the Case business object's list of methods.

- Algorithm:**
- Based on the method name, invoke the corresponding method. For example, if the method to be invoked is `getsite`, invoke `fetchSite()`.
 - Return the result of the operation to the calling function.

Function — `fetchSite()`:

- Algorithm:**
- Get a handle to Case Business object by calling `getMainCBO()`. This handle should point to a specific row in the business object.
 - Use Site property defined in the Case Business object to complete the operation. The site property contains a reference to a Generic business object, which points to the Site that reported the Case. `boCase.getSite()`;



When a query operation is performed on a business object, it returns information about the main business object as well as its child business object. However, for some child business objects, the query mode is disabled by default. The reason is the query might fetch many rows. In that case, the query mode for that child business object must be enabled before doing the query operation.

For example, the email log associated with a case business object can be extracted by using the `EmailLog` property. However, the query mode of this business object is set to Disabled by default. When a query operation is performed in Case business object, the result of the query does not contain any reference to the email log business object. To access the email logs, you have to explicitly set the Query Mode property to `cboSubmitEnabled`.

```
boCase.getEmailLog.setQueryMode(setQueryMode(CboConstants.cboSubmitEnabled);
```

7. Implement the `workflowRPC()` method of the interface. This method performs workflow actions on CBOs. Workflow operations require information about user, queue and WIPBIN objects.

Algorithm:

- Get a handle to the Case Business Object on which the workflow operation has to be performed.
- Get a handle to user, queue and WIPBIN objects.
- Based on the incoming request, invoke a function defined in the wrapper. For example, if the incoming request requires the `Assign` workflow operation to be performed on a Case business object, invoke `performAssign()` defined in the Case Wrapper. This method contains the business logic to perform `Assign` operation.
- Return the result to the calling function.

Function — `performAccept()`:

This method accepts a workflow item into a specified WIPBIN.

- Check whether the incoming request contains any data about the WIPBIN.
- If it has any, accept the workflow business object in the specified WIPBIN.

The `accept` operation can be invoked by using `accept` method of the workflow abstract class:

```
boCase.accept (Base WIPBIN, Boolean isTemporary)
```

where

`boCase` is a handle to the Case business object.

`WIPBIN` represents a handle to the WIPBIN object.

`Boolean` when set to true, the workflow item will be temporarily accepted in the specified WIPBIN. Else move the object to the default WIPBIN.

`boCase.accept ()` returns the status of the operation to calling function.

Function — `performAssign()`:

This method assigns a workflow item to a specified user and places the item in the specified WIPBIN. Input parameters are:

- Handle to a User Object

- Handle to a WIPBIN Object



The `assign` operation can be performed on a workflow object only by the owner of the object. If the logged in user is not the owner, the adapter will not be able to perform the workflow operation on that object.

Before performing a workflow operation the adapter checks whether the logged in user is the owner. If the owner is the logged in user, the adapter invokes the operation to satisfy the request. If not, the adapter changes the ownership of the workflow object and then performs the workflow operation.

The `assign` operation requires information about the WIPBIN and the user. If the incoming request does not have details about the WIPBIN or if the details furnished by the incoming request do not point to any specific WIPBIN, the adapter assigns the workflow object to the user's default WIPBIN.

Algorithm:

- Get a handle to the user and WIPBIN to which the workflow object has to be assigned.
- Check whether the logged in user is the owner of the object. If not change the ownership of the workflow object.
- If the incoming request does not contain information about the user, send a reply back to the calling function specifying that the request does not contain the required information.
- Check whether the request contains information about the WIPBIN. If it points to a specific WIPBIN, assign the workflow object to it. Assign operation can be performed by calling `assign` method of the workflow abstract class.
`boCase.assign (Base User, Base WIPBIN)`
- Else assign the workflow object to the default WIPBIN.
`boCase.assign (Base User)`
- Return the status of the operation to the calling function.

Function — `performDispatch()`:

This method dispatches a workflow item to a specified queue. The input parameter is

- Handle to a Queue Object



The `dispatch` operation can be performed on a workflow object only by the owner of the object. If the logged in user is not the owner, the adapter will not be able to perform the workflow operation on that object.

Before performing a workflow operation, the adapter checks whether the logged in user is the owner. If the owner is the logged in user, the adapter invokes the operation to satisfy the request. If not, the adapter changes the ownership of the workflow object and then performs the workflow operation.

The `dispatch` operation requires information about a `Queue` object to which the workflow object has to be dispatched.

Algorithm:

- Get a handle to the `Queue` to which the workflow object has to be dispatched.
- Check whether the logged in user is the owner of the object. If not change the ownership of the workflow object.
- If the incoming request does not contain information about the queue, send a reply back to the calling function specifying that the request does not contain the required information.
- Else dispatch the workflow object to the specified queue
`boCase.dispatch (Base Queue)`
- Return the status of the operation to the calling function.
- Implement the `customizedRPC()` method of the interface. This method calls a function based on the user request, which then performs the action defined in it. The adapter supports customized operation on Case Business Object. If you want to implement any functionality which the adapter doesn't support, you have to write a method implementing your requirements and that method can be invoked as and when required.

Example: The adapter supports a method `insert2CaseHistory`, which adds a new row in a table called `table_tibco_ex_case_history` based on the field values that come with the incoming Case object.

Function — `performForward()`:

This method forwards a workflow item from its current queue to a specified queue.

The input parameter is:

- Handle to a Queue Object



Forward operation can be performed on a workflow object only by the owner of the object. The object must also be in a queue possessed by the owner. If the logged in user is not the owner, the adapter will be unable to perform the workflow operation on that object.

Before performing a workflow operation, the adapter checks whether the logged in user is the owner. If the owner is same as logged in user, the adapter invokes the operation to satisfy the request. If not, the adapter changes the ownership of the workflow object and also changes the queue in which the object resides to the one associated with the logged in user. It then performs the workflow operation.

Algorithm:

- Get a handle to the Queue to which the workflow object has to be forwarded.
- Check whether the logged in user is the owner of the object. If not, change the ownership of the workflow object.
- Change the queue associated with the object.
- If the incoming request does not contain information about the queue, send a reply back to the calling function specifying that the request does not contain the required information
- Else forward the workflow object to the specified queue. Forward operation can be performed by invoking forward method of the abstract workflow class.
`boCase.forward(Base Queue)`
 where
`boCase` is a handle to the Case business object.
`Queue` is a handle to the Queue object.
- Return the status of the operation to the calling function.

Function — performMove():

This method moves a workflow item from its current WIPBIN to a specified WIPBIN.

The input parameters are:

- Handle to a WIPBIN Object



The `move` operation can be performed on a workflow object only by the owner of the object. If the logged in user is not the owner, the adapter does not perform the workflow operation on that object.

Before performing a workflow operation, the adapter checks whether the logged in user is the owner. If the owner is the logged in user, the adapter invokes the operation to satisfy the request. If not, the adapter changes the ownership of the workflow object and then performs the workflow operation.

The `move` operation requires information about a WIPBIN object to which the workflow object has to be moved. If the incoming request does not possess information about the WIPBIN object, the workflow object will be forwarded to the default WIPBIN.

Algorithm:

- Get a handle to the WIPBIN to which the workflow object has to be moved.
- Check whether the logged in user is the owner of the object. If not change the ownership of the workflow object.
- If the incoming request contains information about the WIPBIN, move the object to the specified WIPBIN. Move operation can be performed by using `move` method of the workflow abstract class.

`boCase.move(Base WIPBIN)`

where

`boCase` is a handle to the Case business object.

WIPBIN represents a handle to the WIPBIN object.

- Else move the workflow object to the default WIPBIN
- `boCase.Move()`
- Return the status of the operation to the calling function.

Function — performReject():

This method rejects a workflow item from the current queue. This operation does not take any input.



The reject operation can be performed on a workflow object only by the owner of the object. The object must be in a queue possessed by the owner. If the logged in user is not the owner, the adapter does not perform the workflow operation on that object.

Before performing a workflow operation, the adapter checks whether the logged in user is the owner. If the owner is the logged in user, the adapter invokes the operation to satisfy the request. If not, the adapter changes the ownership of the workflow object and also changes the queue in which the object resides to the one associated with the logged in user.

Before performing a reject operation, RejectMsgLog business object has to be updated. Use RejectMsg property to update the RejectMsgLog business object. After updating it, perform reject workflow operation.

Algorithm:

- Check whether the logged in user is the owner of the object. If not change the ownership of the workflow object.
- Change the queue associated with the object.
- Update the RejectMsgLog business object
- Reject the workflow object. This can be done by using reject method of the workflow abstract class

```
boCase.reject(Base Queue)
```

where

boCase is a handle to the Case business object.

Queue is a handle to the Queue object.

Return the status of the operation to the calling function.

Function — performYank():

This method yanks a workflow item and places it in the current user's default WIPBIN. It does not take any parameters.

Algorithm:

- The Yank operation can be invoked by using yank method of the workflow abstract class.
- ```
boCase.yank()
```
- Return the status of the operation to calling function.
  - Implement the customizedRPC() method of the interface. This method calls a function based on the user request, which performs the action defined in it. The adapter

supports customized operation on Case Business Object. If you want to implement any functionality which the adapter does not support, you have to write a method implementing your requirements and invoke that method as and when required.

**Example:** The adapter supports a method `insert2CaseHistory` which adds a new row in a table called `table_tibco_ex_case_history` based on the field values that come with the incoming Case business object.

### Function — `insert2CaseHistory`:

- Algorithm**
- Get a handle to a Base business object pointing to the table `table_tibco_ex_case_history`.
  - Add a new row to the business object using `addNew()`.
  - Set the values for all the fields by extracting the data from the Case business object.
  - Update the field values in the current row of the business object to the database by invoking `update()` method.

## Appendix E    **Testing the Adapter**

This chapter describes how to verify that the adapter is installed and configured properly. Tests to verify importing and updating a site in Amdocs CRM are provided in this chapter.

### Topics

---

- [Testing the Publication Service with ClearBasic Forms, page 248](#)
- [Testing the Publication Service with Web Forms, page 252](#)
- [Testing the Request-Response Functionality with Web Forms, page 253](#)
- [Testing the Request-Response Functionality with ClearBasic Forms, page 254](#)

## Testing the Publication Service with ClearBasic Forms

---

Five ClearBasic files are provided along with the adapter to test the Publication service functionality. Before you run the Publication Service, you must import all ClearBasic files using Amdocs CRM tools.

The ClearBasic forms are listed below and are available in the `samples/testPublisher` directory.

- `tibco616.cbs`
- `tibco712.cbs`
- `tibco717.cbs`
- `tibco624.cbs`
- `tibco672.cbs`

You must use Amdocs CRM client-side tools to import the ClearBasic forms into the Amdocs CRM system to be used by the Publication Service.

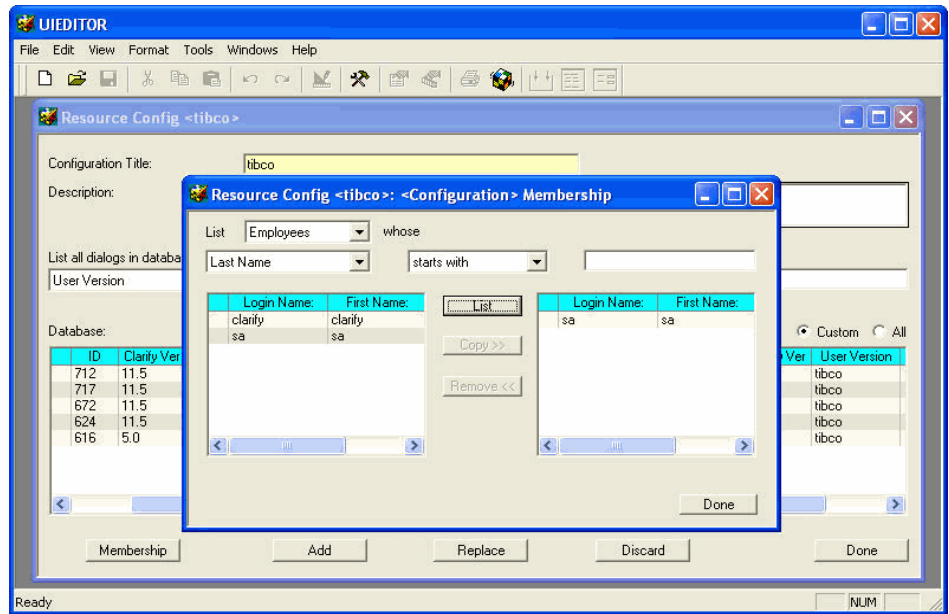
See `readme.txt` in the `samples/testPublisher` directory for additional information.

### Importing ClearBasic Files

The installation procedure assumes you are familiar with the User Interface Editor and ClearBasic Exchange (cbex). For more information about these tools, see the *User Interface Editor Guide* and *ClearBasic's Programmer's Guide*.

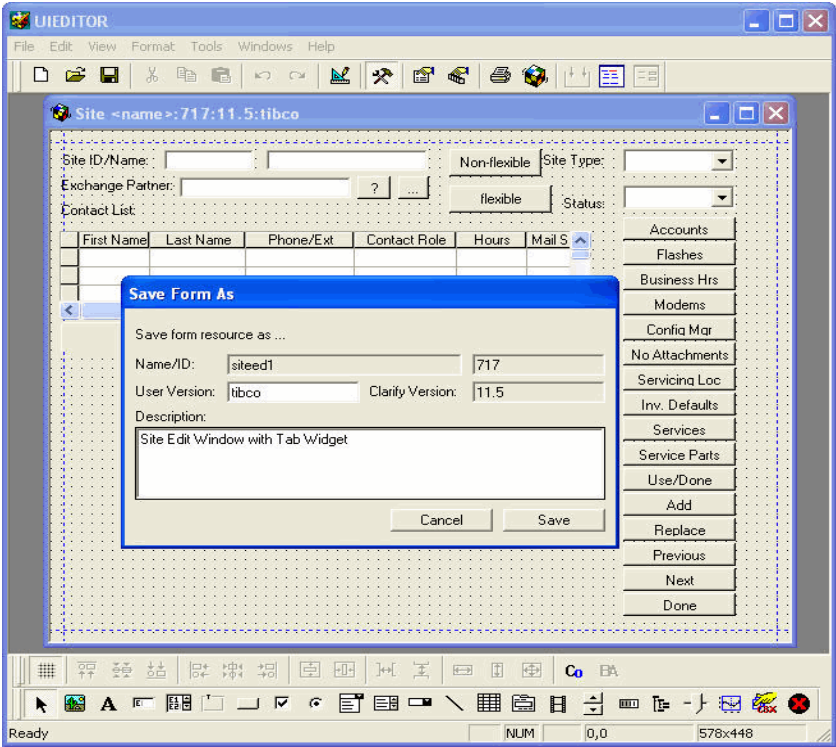
1. Use the UEditor to create a Resource Configuration for those users who will publish data from your forms.
2. Add those users to the Resource Configuration.

Figure 24 Add Users to Resource Configuration



3. Use the UIeditor to create a customized version of the form for which TIBCO customized code is provided in the `samples/testPublisher` directory as separate `cbs` files.

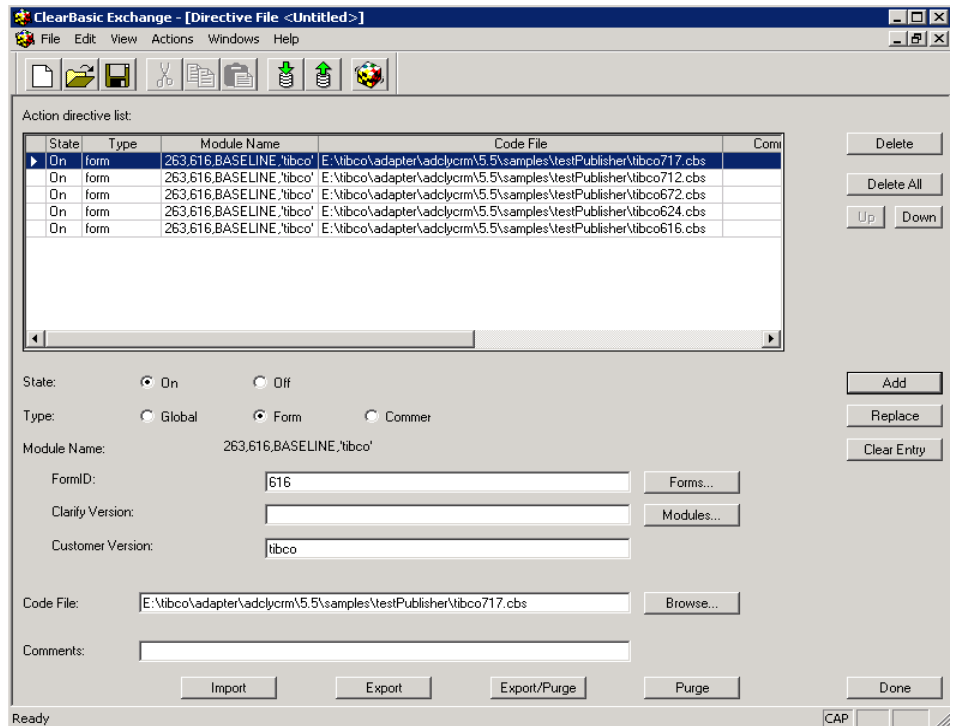
Figure 25 Create Customized Form



- 4. Add these forms to the Resource Configuration you created in [step 1](#).
- 5. Import the forms into the database by entering the command:  
`cbex -dir examples.cbi`

The following Directive File window appears.

Figure 26 Directive window



6. Verify that the code file path is correct. If it is correct, go to [step 8](#).
7. If the code file path is not correct, use the **Browse** button to select the correct path. Then click the **Replace** button to replace the previous path. Continue with [step 8](#).
8. Click the **Import** button.



If the Amdocs CRM system already has ClearBasic customization on the selected forms, you must merge the script provided by TIBCO with your customization.

## Testing the Publication Service with Web Forms

---

Two JSP files are provided along with the adapter to test the Publication Service functionality. Before you run the Publication Service, you must put the two JSP files into the eBusiness Framework installation.

JSP forms are:

- OrderSubmit.jsp
- reg\_do\_action\_1.jsp

See `readme.txt` in the `samples/testPublisher` directory for details.

1. Make sure that your eBusiness framework installation is working fine.
2. Add the `TibcoCboInterface.jar` to the web server classpath.
3. Replace the `OrderSubmit.jsp` in the `eOrderJSP` folder of the eBusiness Framework installation with the `OrderSubmit.jsp` file given in the adapter package.
4. Replace the `reg_do_action_1.jsp` in the `PortalJSP` folder eBusiness Framework installation with the `reg_do_action_1.jsp` file given in the adapter package.
5. Set the value of `iloginRole` variable to `1` in the `Portal.jsp` file in the eBusiness Framework installation.
6. Test the creation of the order.

### To test the creation of the order:

1. Run the eOrder application and create an order.
2. Fill all the required information.
3. Click the **Purchase** button. This will publish the purchase order along with some related objects on the TIBCO Rendezvous.
4. You can view the published message on the adapter console.

### To test the creation of a case:

1. Run the Customer Portal application and move to the B2B Admin User Registration page..
2. Fill all the required information.
3. Click the **SubmitToCSR** button to publish a new case along with its related objects on the TIBCO Rendezvous.
4. You can view the published message on the adapter console.



## Testing the Request-Response Functionality with Web Forms

---

A JSP page, `OrderSubmit.jsp` page, is provided along with the adapter installation to test the Request-Response functionality. Place the `ordersubmit.jsp` into the eBusiness Framework installation.

See `readme.txt` in the `samples/testPublisher` directory for details.

1. Make sure that your eBusiness framework installation is working fine.
2. Add the `RPCGlobalModule.jar` to the web server classpath.
3. Replace the `OrderSubmit.jsp` in the `eOrderJSP` folder of the eBusiness framework installation with the `OrderSubmit.jsp` file given in the adapter package.
4. Test the creation of the order.

### To test the creation of the order:

1. Run the eOrder application and create an order.
2. Fill all the required information and click the **Submit Order** button. This will publish the purchase order along with some related objects on the TIBCO environment.
3. You can view the published message on the adapter console.

## Testing the Request-Response Functionality with ClearBasic Forms

---

A ClearBasic form, `RPCSampleSite.cbs`, is provided along with the adapter installation to test the Request-Response functionality. Using the example, set up one of the Amdocs CRM forms along the lines illustrated in the example. The location of the sample ClearBasic form is given next.

`TIBCO_HOME/adapter/adclycrm/version_num/samples/testRPCClient`

1. Before running the test, ensure that you have completed certain prerequisites. See the following sections for details:
  - [Prerequisites for Setting Up Request-Response Invocation Functionality using Amdocs ClearBasic Forms in a Flexible Deployment on page 103](#)
  - [Prerequisites for Setting Up Request-Response Invocation Functionality Using Amdocs ClearBasic Forms in a Non-Flexible Deployment on page 106](#)
2. Make sure that your eBusiness framework installation is working fine.
3. Add the global RPC module to the Amdocs CRM Applications.
4. Add the code to send the TIBCO Rendezvous message in the form.  
Test for the message you have configured to be published by listening (using `tibrvlisten` or `rvstream`) on that subject.

## Appendix F **Frequently Asked Questions**

This appendix contains answers to some frequently asked questions.

### Topics

---

- [Frequently Asked Questions, page 256](#)

## Frequently Asked Questions

---

### Can I bring up TIBCO Designer from a Unix command-line?

TIBCO Designer is a GUI based tool and a Unix GUI environment is mandatory to run it.

### When starting the adapter, what if the repository is not found?

Start the repository server before starting the adapter.

If you are starting a remote repository, ensure that TIBCO Administrator is installed on the remote location.

Ensure that a properly configured .dat file is available in the path specified (local or remote). Ensure that the RepoUrl has been specified accurately in the adapter's .tra file.

### Why does the adapter startup fail?

Ensure that the RepoUrl syntax has been specified accurately in the adapter's .tra file.

Ensure that the path specified for the .tra file is correct.

### Why does the adapter startup fail, even after specifying the appropriate DAT file?

You must start the repository server before you start the adapter. If it is a remote repository, ensure that the RepoUrl syntax has been specified accurately in the adapter's .tra file. Ensure that the path specified for the .tra file is correct.

### When saving an adapter configuration to the project, why does it result in error messages indicating network problems?

If your installation is a stand-alone installation, check if the *TIBCO\_HOME\TRA\bin* directory is present in your PATH settings.

### When saving an adapter configuration to the project, if an error occurs where is it logged?

TIBCO Designer error messages are logged to the files *stderr.log* and *designer.log* under the *TIBCO\_HOME/Designer/version\_num/logs* directory.

### When an error occurs in a Subscription Service adapter service, where is it displayed?

If the request comes with a reply subject, then the Subscription Service sends back a reply with the exception message as a part of the reply.

Errors that occur in a subscription operation are logged to a trace file. The log file path and name is set in the .tra file corresponding to the adapter instance. All logs are sent to *TIBCO\_HOME/adapter/adclycrm/version\_num/logs* unless otherwise specified.

**Why does the adapter fail to respond to a request?**

The subject name may be inconsistent. The subject name to which the adapter listens may be different from that of the subject name of the client.

**Why does the adapter fail to respond to a request after successfully receiving it?**

The adapter may fail to respond due to various reasons like errors resulting from class mismatch, records not being available in the target application or, connectivity problems with the target applicatio









# Index

## A

adapter instance fields [18](#)  
 Adapter Termination Criteria [22](#)  
 add TIBCO Staging tables to Amdocs CRM database [61](#)  
 add users to resource configuration [259](#)  
 adding adapter services [55](#)  
 agents [140](#)  
 alerts [140](#)  
 Amdocs CRM Business Objects (CBOs) [3](#)  
 Atomicity [4](#)  
 Auto-Discovery process, TIBCO Hawk [142](#)  
 available microagents [147](#)

## B

B2B Admin User Registration (View 1) [262](#)  
 B2B Admin User Registration (View 2) [262](#)

## C

changes from the previous release of TIBCO ActiveMatrix  
     Adapter for Amdocs CRM Configuration and  
     Deployment [xvi](#)  
 changing the polling interval and the update interval [23](#)  
 Class Microagent Name field, adapter [30](#)  
 Classic-LAN and Web Client [3](#)  
 ClearBasic forms [106](#)  
 ClearBasic functions [64](#)  
 ClearBasic Global file [64](#)  
 ClearBasic script [65](#)  
 ClearBasic script coding conventions [67](#)  
 command line arguments to start the adapter [158](#)

commands  
     cbex [62](#), [260](#)  
     -dir [260](#)  
 configuration properties about TIBCO Rendezvous ses-  
     sion, retrieving through TIBCO Hawk [166](#)  
 configure the adapter [8](#)  
 configuring a remote adapter [136](#)  
 configuring multiple Publication service endpoints [37](#)  
 Connection Factory Type [34](#)  
     Queue [34](#)  
     Topic [34](#)  
 Convenient Configuration Tool [4](#)  
 create customized form [260](#)  
 custom JavaBean invocation [87](#)  
 customer support [xx](#)  
 customized operations [100](#)

## D

Database Name [20](#), [21](#), [21](#)  
 Database Port [21](#)  
 Database Server Name [20](#), [21](#)  
 defining a TIBCO Hawk session [129](#)  
 Delayed Acknowledgement [4](#)  
 Delivery Mode [33](#)  
     Durable [34](#)  
     Non-durable [34](#)  
     Non-Persistent [34](#)  
     Persistent [33](#)  
 Directive window [261](#)  
 DTA Transport [20](#), [20](#)

## E

ENV\_NAME [xviii](#)

error message listing [190](#)  
 Error Publisher [91](#)  
 Example of an Input Schema - Query Operation [96](#)  
 examples  
   user exit for the Subscriber [234](#)

## F

Fetching Multiple Rows [3](#)  
 Flexible deployment [106](#)  
 frequently asked questions [268](#)  
 function  
   fetchSite() [249](#)  
   insert2CaseHistory [255](#)  
   performAccept() [249](#)  
   performAssign() [250](#), [250](#)  
   performDispatch() [251](#)  
   performForward() [252](#)  
   performMove() [253](#)  
   performReject() [254](#)  
   performYank() [254](#)

## G

General Tab [24](#)  
 generic configuration properties, retrieving through TIBCO  
   Hawk [158](#)  
 Global Java Class [69](#)  
 global variables [130](#)

## H

how the adapter implements user exits [233](#), [233](#)  
 how the Publication service works [60](#)  
 how the Publication service works for Amdocs CRM Web  
   Client [69](#)  
 how the Request-Response Invocation service works [106](#)  
 how the Request-Response service work [95](#)  
 how the Subscription service works [86](#)

## I

import ClearBasic Global files [62](#)  
 importing ClearBasic files [258](#)  
 incoming message formats for Query operations [96](#)  
 Insert and Update Operations [3](#)  
 Interval Between Reconnect Attempts (milliseconds) [22](#)  
 Invoking Custom JavaBean [3](#)  
 invoking microagent methods [144](#)

## J

JMS [3](#)  
 JSP coding conventions [72](#)

## L

ledger files, retrieving information through TIBCO  
   Hawk [176](#)  
 limitations of the Publication service [82](#)  
 Log File field, adapter [26](#)  
 Log to Standard field, adapter [26](#)

## M

Maximum Number of Reconnect Attempts [22](#)  
 Message Transports [32](#)  
 microagent methods supported [147](#)  
 Microagent Session field, adapter [30](#)  
 modifying the Design-time adapter Rendezvous connection  
   properties [128](#)

## N

Non-Flexible deployment [110](#)  
 Number of Reconnect Attempts Before Suspending  
   Impacted Service(s) [22](#)

**O**

- outgoing message formats for customized operations 101
- overview
  - adapter instance 16
  - TIBCO Hawk 140
  - user exits 232
  - wrapper classes 238

**P**

- Parent-child Schemas 3
- Password 21
- pre-requisites for setting up Request-Response Invocation
  - functionality
  - using Clarify ClearBasic forms in a Flexible
    - deployment 107
  - using Clarify ClearBasic forms in a Non-Flexible
    - deployment 110
  - using Clarify Web forms in a Web Client
    - deployment 115
- Publication service
  - features 56
  - fields 36
  - overview 56

**Q**

- QoS
  - Certified 33
  - Distributed Queue 33
  - Reliable 32
- Quality of Service 32, 32
- Query operations 95

**R**

- Remember Password 20
- Reply subject 91

- Request-Response Limitations, workflow operation 100
- Request-Response Invocation Service
  - Fields 49
  - Overview 104, 104
  - Using Flexible Deployment 104, 104
  - Using the Web Client Deployment 106
- Request-Response service
  - features 94
  - fields 45
  - overview 94
- Request-Response Service Features 94
- Request-Response Service Transport Tab with TIBCO
  - Rendezvous 46, 51
- resetThreadStatistics() 175

**S**

- Sample Code
  - Used for a Flexible Deployment 108
  - Used for a Non-Flexible Deployment over JMS 112
  - Used for a Non-Flexible Deployment over TIBCO
    - Rendezvous 111
  - Used for a Web Client Deployment 117
- Show Startup Banner 29
- Simple Single CBO Query Operation 3
- Starting TIBCO Hawk Software 141, 141
- Subscription Service
  - Features 84, 84
  - Fields 41
  - Functionality 84, 84
  - Overview 84, 84
- Subscription Service Tabs
  - Transport 40
- substitution 130
- support, contacting xx
- Supported Request-Response Operations 94, 94

**T**

- table\_tibco\_message\_fields table 185, 185
- table\_tibco\_message\_queue table 184, 184

technical support [xx](#)

Termination Subject or Topic [24](#)

Testing the Publication Service

with ClearBasic Forms [258](#)

with Web Forms [262](#)

Testing the Request-Response Functionality

with ClearBasic Forms [265](#)

with Web Forms [264](#)

testPublisher [258](#), [258](#), [262](#)

TIBCO ActiveMatrix BusinessWorks [4](#)

TIBCO Hawk

enterprise monitor components [140](#)

TIBCO Hawk methods

getConfig [158](#)

getRvConfig [166](#)

getStatus [167](#)

reviewLedger [176](#)

TIBCO Rendezvous, retrieving configuration through

TIBCO Hawk [166](#)

TIBCO Staging Tables [56](#)

TIBCO\_HOME [xviii](#)

Wire Formats [34](#)

ActiveEnterprise Message [35](#)

Rendezvous Message [35](#)

XML Message [35](#)

workflow operation incoming message formats [96](#)

Workflow Operations [3](#), [97](#), [97](#)

Wrapper Class and TibcoCBOInterface [246](#), [246](#)

Writing Wrapper Class for Case Business Object [247](#), [247](#)

## U

user exits for pre-processing of Subscription service event

message [233](#)

User Name [20](#)

UserExit interface [234](#)

## V

variable substitution [130](#)

variables [130](#)

## W

Web Client Deployment [114](#), [114](#)