# BEA WebLogic Adapter for FIX

## User Guide

Release 7.0.3
Document Date: April 2003

## Copyright

## Restricted Rights Legend

Information in this document is subject to change without notice and does not represent a commitment on the part of BEA Systems. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. FURTHER, BEA Systems DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE SOFTWARE OR WRITTEN MATERIAL IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE.

## Trademarks or Service Marks

**BEA WebLogic Adapter for FIX User Guide**

| Part Number | Date |
|---|---|
| N/A | April 2003 |

# Table of Contents

## 1. Introducing the BEA WebLogic Adapter for FIX

## 2. Metadata, Schemas, and Repositories

## 3. Creating and Configuring an Event Adapter

## 4. Creating and Configuring a Service Adapter

## 5. Using Tracing

## A. Supported Messages

# About This Document

The *BEA WebLogic Adapter for FIX User Guide* is organized as follows:

- Chapter 1, “Introducing the BEA WebLogic Adapter for FIX,” provides an overview of the adapter’s benefits and processes.
- Chapter 2, “Metadata, Schemas, and Repositories,” describes metadata, how to name a schema repository and the schema manifest, how to create a schema, how to store directory and template files for transformations.
- Chapter 3, “Creating and Configuring an Event Adapter,” describes how to create, configure, and test an event adapter.
- Chapter 4, “Creating and Configuring a Service Adapter,” describes how to create, configure, and test a service adapter.
- Chapter 5, “Using Tracing,” describes how to use tracing.
- Appendix A, “Supported Messages,” describes supported messages.

## What You Need to Know

This document is written for system integrators who develop client interfaces between FIX and other applications. It describes how to use the BEA WebLogic Adapter for FIX and how to develop application environments with specific focus on message integration. It is assumed that readers know Web technologies and have a general understanding of Microsoft Windows and UNIX systems.

# Related Information

The following documents provide additional information for the associated software components:

- *BEA WebLogic Adapter for FIX Installation and Configuration Guide*
- BEA WebLogic Adapter for FIX Release Notes
- BEA WebLogic Server installation and user documentation, which is available at the following URL:

  `http://edocs.bea.com/more_wls.html`
- BEA WebLogic Integration installation and user documentation, which is available at the following URL:

  `http://edocs.bea.com/more_wli.html`

# Contact Us!

Your feedback on the BEA WebLogic Adapter for FIX documentation is important to us. Send us e-mail at docsupport@bea.com if you have questions or comments. Your comments will be reviewed directly by the BEA professionals who create and update the BEA WebLogic Adapter for FIX documentation.

In your e-mail message, please indicate which version of the BEA WebLogic Adapter for FIX documentation you are using.

If you have any questions about this version of the BEA WebLogic Adapter for FIX, or if you have problems using the BEA WebLogic Adapter for FIX, contact BEA Customer Support through BEA WebSupport at www.bea.com. You can also contact Customer Support by using the contact information provided on the Customer Support Card, which is included in the product package.

When contacting Customer Support, be prepared to provide the following information:

- Your name, e-mail address, phone number, and fax number

- Your company name and company address
- Your machine type and authorization codes
- The name and version of the product you are using
- A description of the problem and the content of pertinent error messages

# Documentation Conventions

The following documentation conventions are used throughout this document.

| Convention | Item |
|---|---|
| **boldface text** | Indicates terms defined in the glossary. |
| Ctrl+Tab | Indicates that you must press two or more keys simultaneously. |
| *italics* | Indicates emphasis or book titles. |
| `monospace text` | Indicates code samples, commands and their options, data structures and their members, data types, directories, and file names and their extensions. Monospace text also indicates text that you must enter from the keyboard.<br>*Examples*:<br>`#include <iostream.h> void main ( ) the pointer psz`<br>`chmod u+w *`<br>`\tux\data\ap`<br>`.doc`<br>`tux.doc`<br>`BITMAP`<br>`float` |
| **`monospace boldface text`** | Identifies significant words in code.<br>*Example*:<br>`void` **`commit`** `( )` |

| Convention | Item |
|---|---|
| *monospace italic text* | Identifies variables in code.<br>*Example*:<br>`String` *`expr`* |
| UPPERCASE TEXT | Indicates device names, environment variables, and logical operators.<br>*Examples*:<br>LPT1<br>SIGNON<br>OR |
| `{ }` | Indicates a set of choices in a syntax line. The braces themselves should never be typed. |
| `[ ]` | Indicates optional items in a syntax line. The brackets themselves should never be typed.<br>*Example*:<br>`buildobjclient [-v] [-o name ] [-f` *`file-list`*`]...`<br>`[-l` *`file-list`*`]...` |
| \| | Separates mutually exclusive choices in a syntax line. The symbol itself should never be typed. |
| `...` | Indicates one of the following in a command line:<br>■ That an argument can be repeated several times in a command line<br>■ That the statement omits additional optional arguments<br>■ That you can enter additional parameters, values, or other information<br>The ellipsis itself should never be typed.<br>*Example*:<br>`buildobjclient [-v] [-o name ] [-f` *`file-list`*`]...`<br>`[-l` *`file-list`*`]...` |
| .<br>.<br>. | Indicates the omission of items from a code example or from a syntax line. The vertical ellipsis itself should never be typed. |

# 1 Introducing the BEA WebLogic Adapter for FIX

This section provides an overview of the adapter's benefits and processes. It includes the following topics:

## Introduction

The BEA WebLogic Adapter for FIX takes the chore out of integrating securities-related information with enterprise applications, such as order management, and third-party trading partners. The BEA WebLogic Adapter for FIX enables developers to easily read, write, and enrich FIX transactions, while shielding developers from the complexities of the computing environment. The BEA WebLogic Adapter for FIX is well suited for business-process improvement initiatives. It helps replace manual business processes with automatic transaction handling and application-to-application integration, thus dramatically reducing error rates and information lag time. Further, because the BEA WebLogic Adapter for FIX simplifies transaction handling for the developer, it significantly reduces the time, cost, and skill level required for integration projects.

The Financial Information Exchange (FIX) protocol is a message standard designed to facilitate the electronic exchange of securities-related information between brokerage houses, Electronic Communication Networks (ECNs), custodians, and banks. FIX was originally defined for use in supporting U.S. domestic equity trading with message traffic flowing directly between principals. As the protocol evolved, the FIX specification was expanded to support limited cross-border and fixed income trading. Similarly, the protocol was expanded to allow third parties to participate in the delivery of messages between trading partners. As subsequent versions of FIX are released, it is expected that functionality will continue to expand.

FIX was written to be independent of any specific communications protocol or physical medium chosen for electronic data delivery. The BEA WebLogic Adapter for FIX provides support for FIX protocol levels 4.1, 4.2, and 4.3. The BEA WebLogic Adapter for FIX fully supports the full 39 message types defined in FIX 4.2 specification that support pre-trade, trade, post-trade, and clearing/settlement processes in the securities industry.

Combined with BEA WebLogic Integration, the BEA WebLogic Adapter for FIX provides a unique graphical studio and business process management for defining integration rules and mapping the transformation and workflows for FIX integration with enterprise systems and external trading partners. The BEA WebLogic Adapter for FIX can also dynamically format and transform FIX messages into FIXML for internal processing. This enables organizations to use FIXML for communications with internal systems and FIX messages to communicate across networks with trading counter-parties. Even complex workflows for straight-through processing (STP), a major initiative in the securities industry, are simply managed by teaming the BEA WebLogic Adapter for FIX with other components from the BEA WebLogic Integration platform.

The BEA WebLogic Adapter for FIX has both a FIX listener and a FIX emitter, so Studio applications can be integrated as both a sender and receiver of FIX messages.

# Key Features

The BEA WebLogic Adapter for FIX has the following features

- FIX messages can be customized through XML files
- Supports pre- and post-encryption of messages using DES and PGP/DES/MD5
- Supports configuration of events and services in WebLogic Integration
- Routing and Transformation of FIX messages based on business rules defined in the WebLogic Integration Business Process Management Studio (for example routing an incoming indication of interest (IOI) for a particular security to a particular trading desk)
- Automated processing of FIX message reduces clerical errors and increases productivity
- Automated real-time information flow and database integration of FIX messages reduces market risk

An example of FIX message is shown below:

```
8=FIX.4.2_9=73_35=D_11=555555555_21=2_55=BEAS_54=2_60=20020712-15
:15:11_54=2_40=1_38=1000_10=122_
```

The XML representation of this message is:

```
<Fix>
<BeginString>FIX.4.2</BeginString>
<BodyLength>73</BodyLength>
<MsgType>D</MsgType>
<ClOrdID>555555555</ClOrdID>
<HandlInst>2</HandlInst>
<Symbol>BEAS</Symbol>
<Side>2</Side>
<TransactTime>20020711-15:15:11</TransactTime>
<Side>2</Side>
<OrdType>1</OrdType>
<OrderQty>1000</OrderQty>
</Fix>
```

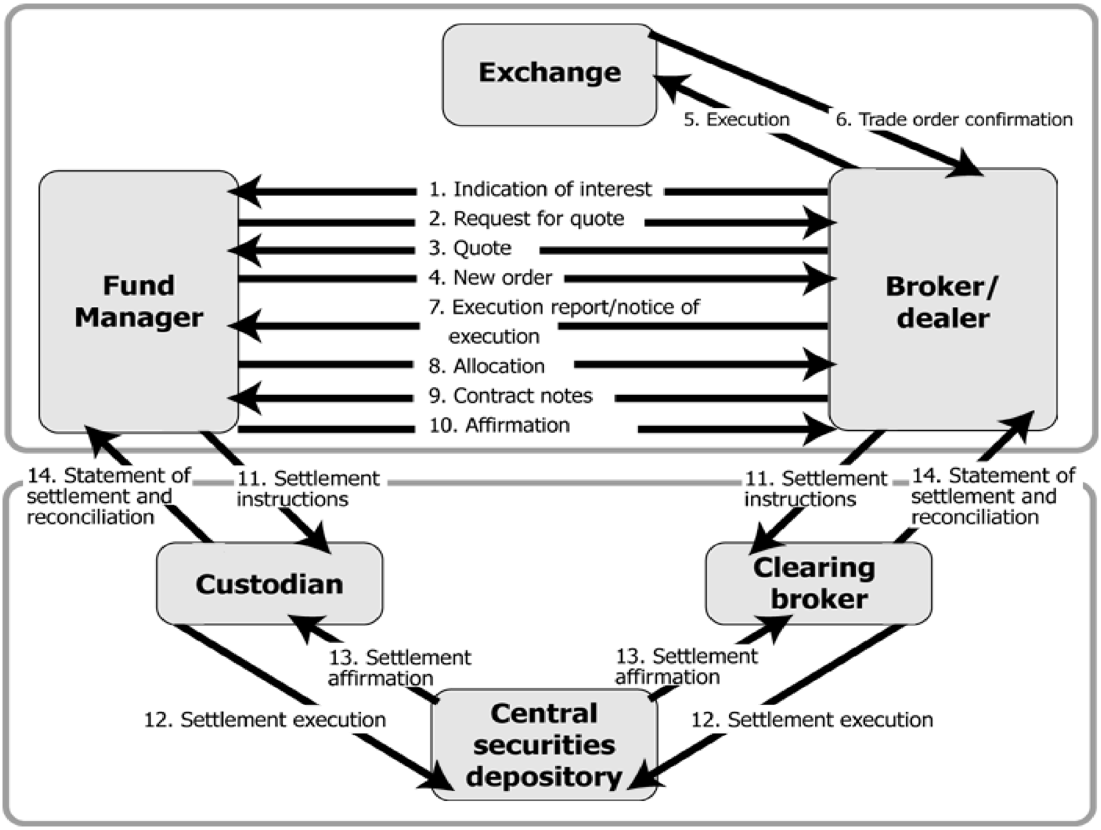
The use of XML as an integration “lingua franca” also promotes integration based on future FIXML standards. FIXML was devised in 1998 by FIX Protocol Ltd., the company responsible for the Financial Information Exchange (FIX) electronic communications protocol, as an XML vocabulary based on the FIX protocol. Its aim is to continue FIX's goal of improving the global trading process by facilitating the exchange of real-time securities transactions. In July 2001, the organization announced plans to team up with SWIFT, which had been working on its own swiftML XML initiative. The collaboration centers around plans to converge their messaging protocols to create an XML-based version of the ISO 15022 protocol for securities message types, which is being developed by the International Standards Organization (ISO). The resultant ISO 15022 XML is to leverage FIX Protocol's expertise in the pre-trade/trade execution domain, and SWIFT's post-trade domain expertise to bring together different parts of the trade life cycle and work through issues hindering straight-through-processing (STP).

BEA provides FIXML support with the BEA WebLogic Adapter for FIX. This adapter supports the non-XML message formats of FIX (4.1, 4.2, 4.3) together with the new ISO 15022 FIXML message formats. The BEA WebLogic Adapter for FIX can also dynamically format and transform FIX messages into FIXML or internal processing. This enables the use of FIXML for internal systems but still uses FIX messages for communication across networks with trading counter-parties.

# BEA WebLogic Adapter for FIX and Workflows

BEA WebLogic Integration when combined with the BEA WebLogic Adapter for FIX allows the implementation of workflow processes for pre-trade and post-trade business processes. The BEA WebLogic Adapter for FIX provides transformation services for FIX messages. The following figure illustrates the workflows involved in processing FIX messages:

**Figure 1-1 Trading Scenario**


Exchange
5. Execution
6. Trade order confirmation
Fund Manager
Broker/ dealer
1. Indication of interest
2. Request for quote
3. Quote
4. New order
7. Execution report/notice of execution
8. Allocation
9. Contract notes
10. Affirmation
14. Statement of settlement and reconciliation
11. Settlement instructions
11. Settlement instructions
14. Statement of settlement and reconciliation
Custodian
Clearing broker
13. Settlement affirmation
13. Settlement affirmation
12. Settlement execution
12. Settlement execution
Central securities depository

# Background of FIX Protocol

The Financial Information eXchange (FIX) protocol is a messaging protocol developed specifically for real-time electronic exchange of securities transactions. FIX is a public-domain, vendor-neutral specification owned and maintained by FIX Protocol, Ltd. The FIX Protocol Ltd. organization strives to improve the global trading process by defining, managing and promoting an open protocol for real-time, electronic communication between industry participants, while complementing industry standards.

FIX was created to support Equities trading, but has been extended to support trading other types of securities, such as Futures, Options, Derivatives, Fixed Income and Foreign Exchange. FIX provides a single, extensible messaging protocol used between traders, brokerages, ECNs and exchanges. FIX furthers the industry's goal of Straight Through Processing and T+1 Trade Settlement.

Historically, security trading involved a great deal of voice communication and paperwork. The transfer process was both time consuming and error prone. Automation of the trading life cycle started with the back office, and has improved the efficiency of the process significantly. However, continual pressures to shorten the trading life cycle, T+3 for example, have caused us to look to the front office for opportunities to improve the trading process. The technology to support electronic trading is now mainstream. However, until recently, the only protocol for formatting securities transactions existed in proprietary form bundled with vendors software. The FIX (Financial Information Exchange) Committee was formed to address this specific need and has developed a standardized message format for describing security transactions.

The FIX protocol is the established standard in electronic communications for sending indications, orders, and executions among major securities firms in the equities market. FIX has grown dramatically as major players continue to transfer their existing and new traffic to it. FIX is also a major player in growing international traffic, especially in Europe. The protocol is establishing itself as a de facto industry standard in the pre-trade equity trading process.  The protocol is now the closest thing the securities industry has to a full-blown standard, and is expected to grow in usage, especially as it becomes a major factor in the industry’s move towards Straight Through Processing.

FIX is flexible, having been used by firms for various functions, from Indications of Interest (IOIs) to administrative messages. FIX is also platform independent, and its benefits have been well reported. In keeping with these benefits, an entire new market inside financial services technology has opened. The Financial Information Exchange (FIX) Protocol is a message standard developed to facilitate the electronic exchange of information related to Equity and Fixed Income transactions. It is intended for use between brokers and institutions wishing to automate communications.

The message protocol, as defined, will support the following electronic conversations:

- Equity order submissions, cancellations and replacements
- Equity execution reporting
- Equity order status

- Equity trade allocation
- Indication of interest communication
- Completed trade advertisements
- Directed email and news messaging

The FIX protocol is defined at two levels; session and application. The session level is concerned with the delivery of data while the application level defines business related data content. This protocol is independent of the telecommunications protocol (X.25, asynch, internet, etc.) and medium chosen for electronic data delivery. In succinct terms, the FIX protocol is currently a specification for only the format data (a.k.a. messages) to be exchanged for a given type of transaction.

The benefits of FIX are:

- Error Reduction: With the introduction of T+3, the opportunity for error correction is significantly reduced. Electronic trade transmission provides a mechanism for reducing the opportunity for human error resulting from audio miscommunication or from erroneous re-keying of trade data.
- More Information: Throughout the life cycle of an order, the Buy-Side can receive interim trade reports without the need for a phone call. This may be particularly helpful in an active market.
- Productivity: Electronic order transmission allows traders to handle higher volumes and use their time more effectively by reducing administrative activity, such as data entry and order status reporting.
- Common Language: The FIX standard provides a common description of trade characteristics so that all participants are speaking the same language. This paves the way for integrating incoming data into internal systems.
- Platform Independent: The FIX protocol is a message format description. It does not require any specific hardware or telecommunication technology. Each participant can use their own hardware platform.
- Access to Buyers: The FIX protocol provides a format for electronically broadcasting indications of interest to Buy-Side firms.
- Access to Markets: Buy-Side firms can electronically transmit DOT orders to the Sales Trader desk where the order can be routed directly to the exchange floor

# Summary

The BEA WebLogic Adapter for FIX integrates your FIX application messages with WebLogic Integration in a fast, easy, and reliable way. You can use the Adapter to exchange XML, non-XML, ASCII, and custom data formats between your FIX resources and WebLogic Integration to provide a tightly integrated and reliable application infrastructure. The BEA WebLogic Adapter for FIX provides:

- Guaranteed asynchronous, bi-directional message interactions between WebLogic Integration and native FIX server destinations.
- Data transfer between a business process running within WebLogic Integration and an FIX Client/Server.
- Service and event adapter integration operations providing end-to-end business process management using XML schemas.

CHAPTER

# 2 Metadata, Schemas, and Repositories

This section explains how metadata for your enterprise information system (EIS) is described, how to name a schema repository and the schema manifest, how to create a schema, and how to store directory and template files for transformations. After the metadata for your EIS is described, you can create and deploy application views using the WebLogic Application View Console.

This section includes the following topics:

- Understanding Metadata
- Schemas and Repositories
- The Repository Manifest

## Understanding Metadata

When you define an application view, you are creating an XML-based interface between WebLogic Integration and an enterprise information system (EIS) or application within your enterprise. The BEA Adapter for FIX is used to define a file based interface to applications within and outside of the enterprise. Many applications or information systems use file systems to store and share data. These files contain information required by other applications, and this information can be fed via the BEA WebLogic Adapter for FIX.

For example, Excel is a widely used application that allows all types of professionals (from fund managers to administrative assistants) to collate information pertinent to their working environment. This information can be shared by other applications using the adapter’s transformation capability, which can convert a worksheet to XML and to other business partners via an EDI stream.

# Schemas and Repositories

You describe all the documents entering and exiting your WebLogic Integration system using W3C XML schemas. These schemas describe each event arriving to and propagating out of an event, and each request sent to and each response received from a service. There is one schema for each event and two for each service (one for the request, one for the response). The schemas are usually stored in files with an `.xsd` extension.

Use the WebLogic Integration Application View Console to access events and services, and to assign a schema to each event, request, and response. Assign each application view to a schema repository; several application views can be assigned to the same repository.

BEA WebLogic Adapters all make use of a schema repository to store their schema information and present it to the WebLogic Application View Console. The schema repository is a directory containing:

- A manifest file that describes the event and service schemas.
- The corresponding schema descriptions.

To work with schemas, you must know how to:

- Name a schema repository.
- Create a manifest.
- Create a schema.

# Naming a Schema Repository

The schema repository has a three-part naming convention:

*session_base_directory*\*adapter*\*connection_name*

- *session_base_directory* is the schema's session base path, which represents a folder under which multiple sessions of schemas may be held.
- *adapter* is the type of adapter (for example, FIX or SAP).
- *connection_name* is a name representing a particular instance of the adapter type.

For example, if the session base path is /usr/opt/bea/bse, the adapter type is FIX, and the connection name is FIXDev, then the schema repository is the directory:

```
/usr/opt/bea/bse/FIX/FIXDev
```

# The Repository Manifest

Each schema repository has a manifest that describes the repository and its schemas. This repository manifest is stored as an XML file named manifest.xml.

The following is an example of a sample manifest file showing relationships between events and services and their schemas.

The manifest file relates documents (through their schemas) to services and events. The manifest exposes schema references to the event relating the required document (via the root tag) to the corresponding schema. Schemas and manifests are stored in the same directory, the repository root of the EIS. The following is an example of the manifest file with a description of the elements.

**Listing 2-1 Sample Manifest File**

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<manifest>
       <connection/>
```

```
        <schemaref name="version_4_1"
                <request root="FIXMLMessage" file="FIXML41.xsd"/>
                <response root="emitStatus" file="emitStatus.xsd"/>
                <event root="FIXMLMessage" file="FIXML41.xsd"/>
        </schemaref>
        <schemaref name="version_4_2">
                <request root="FIXMLMessage" file="FIXML42.xsd"/>
                <response root="emitStatus" file="emitStatus.xsd"/>
                <event root="FIXMLMessage" file="FIXML42.xsd"/>
        </schemaref>
        <schemaref name="version_4_3">
                <request root="FIXMLMessage" file="FIXML43.xsd"/>
                <response root="emitStatus" file="emitStatus.xsd"/>
                <event root="FIXMLMessage" file="FIXML43.xsd"/>
        </schemaref>
</manifest>
```

The manifest has a connection section (which is not used by the BEA WebLogic Adapter for FIX) and a schema reference section, named schemaref. The schema reference name is displayed in the schema drop-down list on the Add Service and Add Event windows in the WebLogic Integration Application View Console. This sample manifest has three schema references or schemaref tags; one for services only, one for events only, and one for a combination of services and events. Events require only one schema, defined by the *event* tag. This relates the root tag of an XML document to a schema in the EIS repository. For services, two schemas are required: one for the document being passed to the service, represented by the *request* tag, and one for the expected *response* document received from the service operation, represented by the **response** tag.

CHAPTER

# 3 Creating and Configuring an Event Adapter

An event adapter is the inbound interface from Financial Information eXchange (FIX) protocol clients to WebLogic Integration Studio.This section describes how to create, configure, and test an event adapter. It includes the following topics:

- Creating an Application View Folder
- Creating an Event Adapter Application View
- Configuring an Event Adapter Application View
- Testing Events Using Application View Console
- Testing Application View Events Using Studio

## Creating an Application View Folder

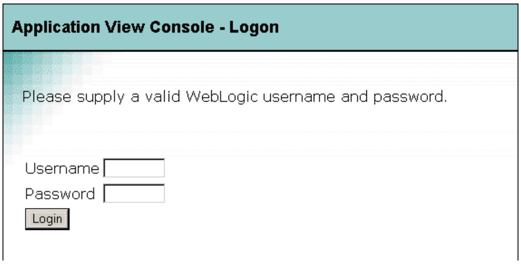
Application views reside within WebLogic Integration. WebLogic Integration provides you with a root folder in which you can store all of your application views; if you wish, you can create additional folders to organize related application views into groups.

To create an application view folder:

1. Log on to the WebLogic Integration Application View Console at //*appserver-host:port*/wlai, where *appserver-host* is the IP address or host name where the WebLogic Integration Server is installed, and *port* is the socket on which the server is listening. The port, if not changed during installation, defaults to 7001.

2. If prompted, enter a user name and password, as shown in the following figure:

**Figure 3-1 Application View Console Logon Window**



Application View Console - Logon

Please supply a valid WebLogic username and password.

Username

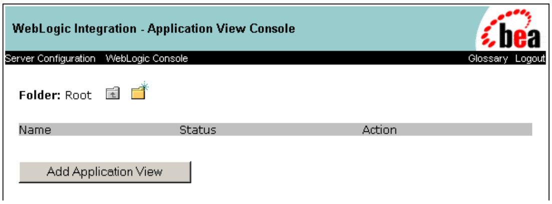
Password

**Note:** If the user name is not `system`, it must be included in the `adapter` group. For more information on adding the administrative server user name to the `adapter` group, see the *BEA WebLogic Adapter for FIX Installation and Configuration Guide*.

3. Click Login.

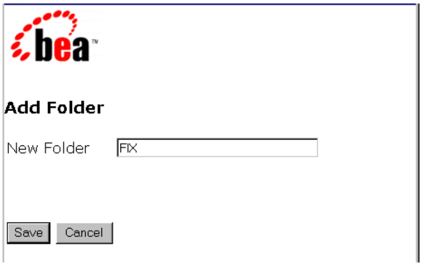
The WebLogic Integration Application View Console opens.

**Figure 3-2 Application View Console Window**



4. Click the new folder icon. The Add Folder page opens:

**Figure 3-3 Application View Console: Add Folder**



Add Folder

New Folder FIX

5. Supply a name for the folder, and then click Save.

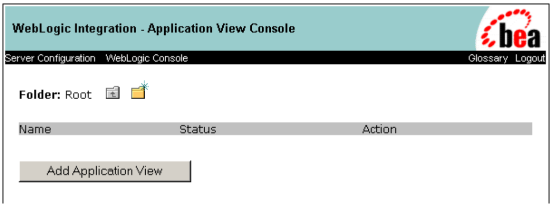
You have finished creating the application view folder.

# Creating an Event Adapter Application View

To create an event adapter application view:

1. Log on to the WebLogic Integration Application View Console at //*appserver-host:port*/wlai, where *appserver-host* is the IP address or host name where the WebLogic Integration Server is installed, and *port* is the socket on which the server is listening. The port, if not changed during installation, defaults to 7001.
2. At the main page, select the desired Application View folder.

**Figure 3-4 Application View Console: Selecting the FIX Folder**



WebLogic Integration - Application View Console

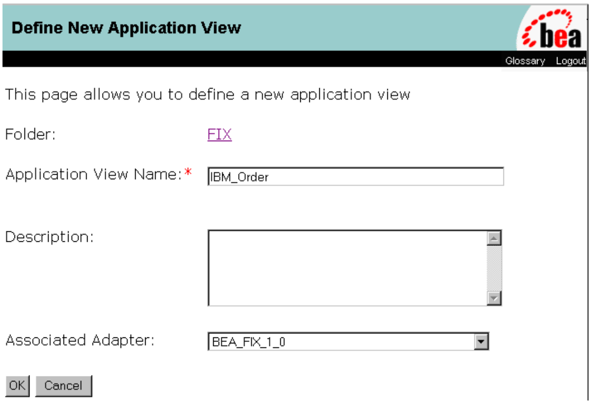
Folder: Root

| Name | Status | Action |
|---|---|---|

3. Click Add Application View. The Define New Application View page opens. An application view enables a set of business processes for this adapter's target EIS application. For more information, see “Defining an Application View” in *Using Application Integration*:

- For WebLogic Integration 7.0, see `http://edocs.bea.com/wli/docs70/aiuser/2usrdef.htm`
- For WebLogic Integration 2.1, see `http://edocs.bea.com/wlintegration/v2_1sp/aiuser/2usrdef.htm`

**Figure 3-5 Application View Console: Define New Application View**



This page allows you to define a new application view

Folder: FIX

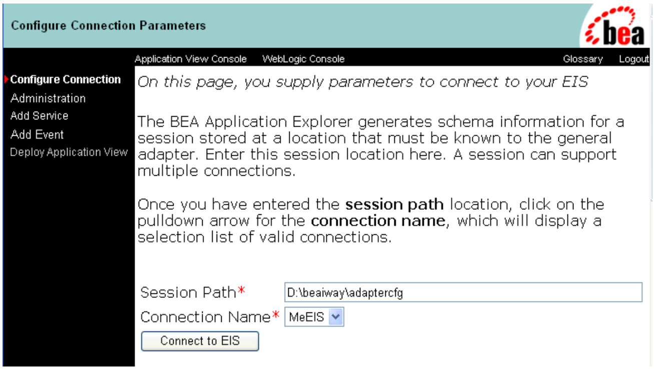
Application View Name:* IBM_Order

Description:

Associated Adapter: BEA_FIX_1_0

4. Type a name and description for the Application View.
5. Select BEA_FIX_1_0 from the Associated Adapter drop-down list.
6. Click OK. The Configure Connection Parameters page opens.

**Figure 3-6 Application View Console: Configure Connection Parameters**



*On this page, you supply parameters to connect to your EIS*

The BEA Application Explorer generates schema information for a session stored at a location that must be known to the general adapter. Enter this session location here. A session can support multiple connections.

Once you have entered the **session path** location, click on the pulldown arrow for the **connection name**, which will display a selection list of valid connections.

Session Path* D:\beaiway\adaptercfg

Connection Name* MeEIS

7. Supply the name of the BEA WebLogic Adapter for FIX session base directory in the Session path box. This directory holds your FIX schema information, and contains the subdirectory FIX/*YourConnectionName*.

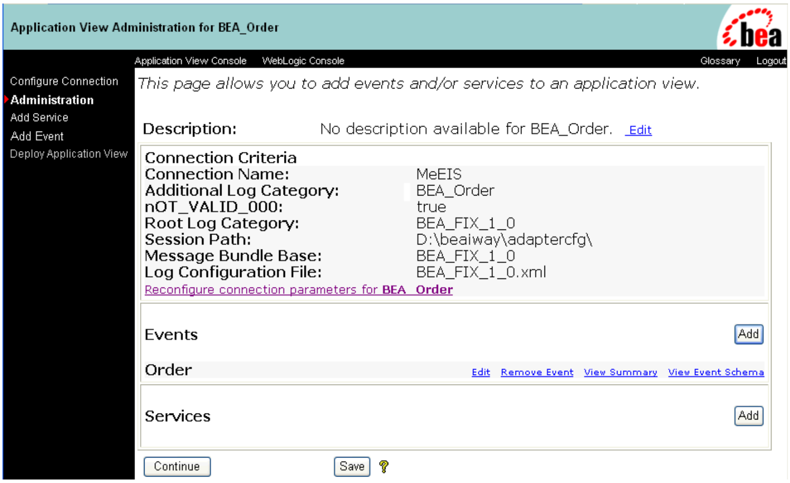
   For example, the session base directory might be d:\bea\bse\sessions\default, with the schema repository—containing a repository manifest and schemas—residing in d:\bea\bse\sessions\default\FIX\IBM.

8. Select the session name—also known as the connection name—from the Connection name drop-down list box.

9. Click Connect to EIS. The Application View Administration page opens.

   Note that you can access the Configure Connection Parameters page (displayed in the previous step) when the application view is not deployed, simply by selecting the Reconfigure connection parameters link. If the application view is deployed, you can access the page by first undeploying the application view.

**Figure 3-7 Application View Console: Application View Administration**



*This page allows you to add events and/or services to an application view.*

Description: No description available for BEA_Order. Edit

Connection Criteria

| | |
|---|---|
| Connection Name: | MeEIS |
| Additional Log Category: | BEA_Order |
| nOT_VALID_000: | true |
| Root Log Category: | BEA_FIX_1_0 |
| Session Path: | D:\beaiway\adaptercfg\ |
| Message Bundle Base: | BEA_FIX_1_0 |
| Log Configuration File: | BEA_FIX_1_0.xml |

Reconfigure connection parameters for **BEA_Order**

10. Click Save.

You have finished creating the application view for the event adapter.

Note that you must add an event, as described in "Configuring an Event Adapter Application View" on page 3-6, before you can deploy the application view.

# Configuring an Event Adapter Application View

After you create an application view, you can add an event to the application view. For more information on creating application views, see "Creating an Event Adapter Application View" on page 3-3. An event adapter application view contains all events that are expected to arrive at an instance of the event adapter. You can add many events to an application view. Each event has a schema for the arriving message (a message is also known as a document). A service should be added for each event that will be used by the application view.

To add an event to, and deploy, an event adapter application view:

1. Log on to the WebLogic Integration Application View Console at //*appserver-host:port*/wlai.

2. Select the folder in which this application view resides, and then select the application view.

   For more information, see "Editing an Application View" in "Defining an Application View" in *Using Application Integration*:

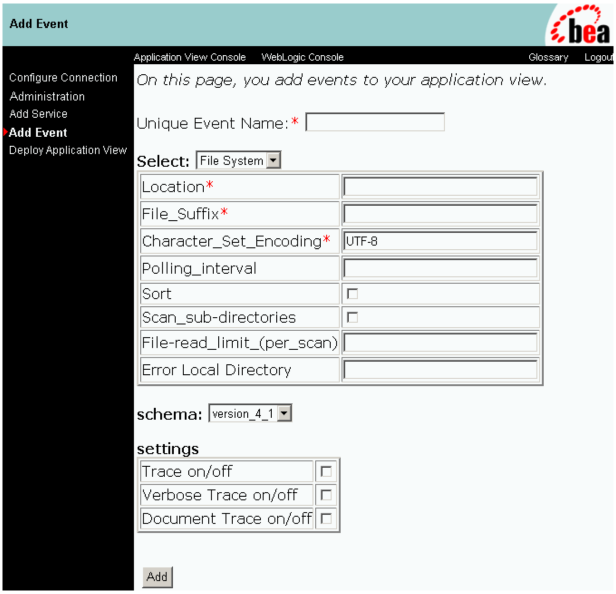
   - For WebLogic Integration 7.0, see `http://edocs.bea.com/wli/docs70/aiuser/2usrdef.htm`

   - For WebLogic Integration 2.1, see `http://edocs.bea.com/wlintegration/v2_1sp/aiuser/2usrdef.htm`

3. In the Administration page of the WebLogic Integration Application View Console, select Add Event. The Add Event page opens:

   Two protocols are available: File and FTP.

**Figure 3-8 Application View Console: File System Add Event**



On this page, you add events to your application view.

Unique Event Name:*

Select: File System

| | |
|---|---|
| Location* | |
| File_Suffix* | |
| Character_Set_Encoding* | UTF-8 |
| Polling_interval | |
| Sort | □ |
| Scan_sub-directories | □ |
| File-read_limit_(per_scan) | |
| Error Local Directory | |

schema: version_4_1

settings

| | |
|---|---|
| Trace on/off | □ |
| Verbose Trace on/off | □ |
| Document Trace on/off | □ |

The following table describes the properties in this window.

**Table 3-1 File System Add Event Properties**

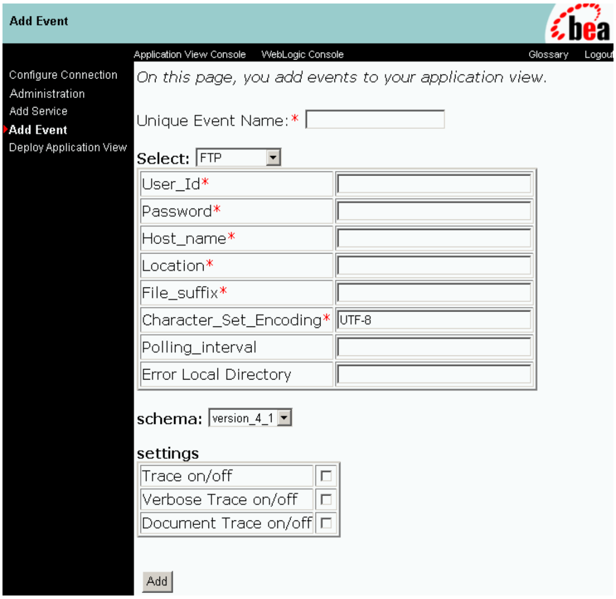
| Property | Description | Type | Sample Value |
|---|---|---|---|
| Location | Port to listen to on for FIX Client messages. This parameter accepts a port number. | string | 3333 |
| File_Suffix | Full path to the fixServer configuration file | string | Fix/fixServer.properties |
| Character_Set_encoding | FIX version | drop down | 4_1, 4_2, 4_3 |
| Polling Interval | The maximum wait interval between checks for new documents. Thehigher this value, the longer the interval, and the fewer system resources that are used. The side effect of a high value is that the worker thread will not be able to respond to a stop command. If timeout is set to 0, the listener will run once and terminate. Default is 2 seconds. | integer/string xxH:xxM:xxS | 1H:2M:3S (1 hour 2 minutes and 3 seconds) |

**Table 3-1 File System Add Event Properties**

| Property | Description | Type | Sample Value |
|---|---|---|---|
| Sort | Sort by arrival - If set, sort incoming documents by arrival time. Maintains sequence, but slows performance. | Boolean (true/false) | |
| Scan_sub-directories | When checked, scans all subdirectories for documents to be processed. | Boolean (true/false) | |
| File-read_limit_(per_scan) | The number of files read per sweep of the File directory location. | integer | |
| Error Local Directory | Directory to which the document being processed is written to in the event of a failure (for instance if the transformation fails or the document does not match the schema). | directory path | |

The following parameters are available for FTP Events

**Figure 3-9 FTP Event Properties**



*On this page, you add events to your application view.*

Unique Event Name:*

Select: FTP

| | |
|---|---|
| User_Id* | |
| Password* | |
| Host_name* | |
| Location* | |
| File_suffix* | |
| Character_Set_Encoding* | UTF-8 |
| Polling_interval | |
| Error Local Directory | |

schema: version_4_1

settings

| | |
|---|---|
| Trace on/off | □ |
| Verbose Trace on/off | □ |
| Document Trace on/off | □ |

**Table 3-2 FTP Event Properties**

| Property | Description | Type | Sample Value |
|---|---|---|---|
| User_Id | User account to use when connecting to the protocol host. | string | |
| Password | Password for the user account to use when connecting to the protocol host. | string | |
| Host_name | Name of host the machine where the listener will contact the service to obtain requests. | string | |
| Location | Port to listen to on for FIX Client messages. This parameter accepts a port number. | string | 3333 |
| File_suffix | Full path to the fixServer configuration file | string | Fix/fixServer.properties |
| Character_Set_Encoding | FIX version | drop down | 4_1, 4_2, 4_3 |
| Polling_Interval | The maximum wait interval between checks for new documents. Thehigher this value, the longer the interval, and the fewer system resources that are used. The side effect of a high value is that the worker thread will not be able to respond to a stop command. If timeout is set to 0, the listener will run once and terminate. Default is 2 seconds. | integer/string xxH:xxM:xxS | 1H:2M:3S (1 hour 2 minutes and 3 seconds) |
| Error Local Directory | Directory to which the document being processed is written to in the event of a failure (for instance if the transformation fails or the document does not match the schema). | directory path | |

4. Select a schema from the drop-down list.

   The outbound FIX messages are sent in FIX format (tag based and not XML based). The schema drop-down list box corresponds to the list of events in the schema repository

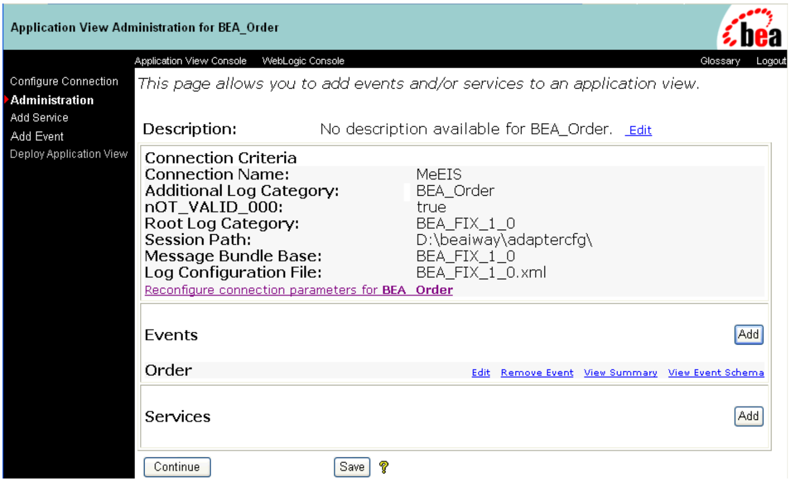
5. Select trace settings, as defined below.

**Table 3-3 Trace Setting Properties**

| Property | Description |
|---|---|
| Trace on/off | Generates a basic trace that displays the input XML (up to 300 bytes) before parsing, and shows the request being processed. For more information about tracing, see Chapter 5, "Using Tracing.". |
| Verbose Trace on/off | Generates a trace that displays configuration parameters used by the adapter. For more information about tracing, see Chapter 5, "Using Tracing.". |
| Document Trace | Generates a trace that displays the input document after it was analyzed and the response document being returned. For more information about tracing, see Chapter 5, "Using Tracing.". |

6. Click Add. The Application View Administration page opens:

**Figure 3-10 Application View Console: Application View Administration**



Application View Administration for BEA_Order

*This page allows you to add events and/or services to an application view.*

Description: No description available for BEA_Order. Edit

Connection Criteria

| | |
|---|---|
| Connection Name: | MeEIS |
| Additional Log Category: | BEA_Order |
| nOT_VALID_000: | true |
| Root Log Category: | BEA_FIX_1_0 |
| Session Path: | D:\beaiway\adaptercfg\ |
| Message Bundle Base: | BEA_FIX_1_0 |
| Log Configuration File: | BEA_FIX_1_0.xml |

Reconfigure connection parameters for BEA_Order

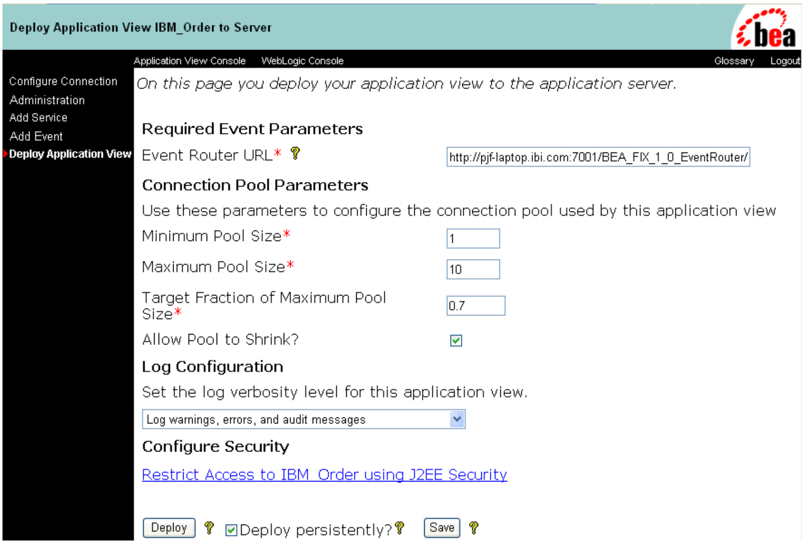
Events

Order

Services

7. Click Continue. The Deploy Application View page opens:

**Figure 3-11 Application View Console: Deploy Application View**



Deploy Application View IBM_Order to Server

*On this page you deploy your application view to the application server.*

Required Event Parameters

Event Router URL* http://pjf-laptop.ibi.com:7001/BEA_FIX_1_0_EventRouter/

Connection Pool Parameters

Use these parameters to configure the connection pool used by this application view

Minimum Pool Size* 1

Maximum Pool Size* 10

Target Fraction of Maximum Pool Size* 0.7

Allow Pool to Shrink? ☑

Log Configuration

Set the log verbosity level for this application view.

Log warnings, errors, and audit messages

Configure Security

Restrict Access to IBM_Order using J2EE Security

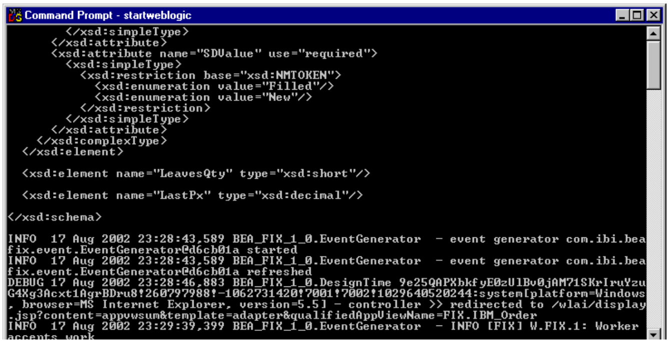
☑ Deploy persistently?

8. Modify event parameters, connection pool parameters, log configuration, and security as necessary.

9. Click Deploy to save and deploy the event adapter.

   In the WebLogic Server Log, you should see the following entries as the event adapter deploys:

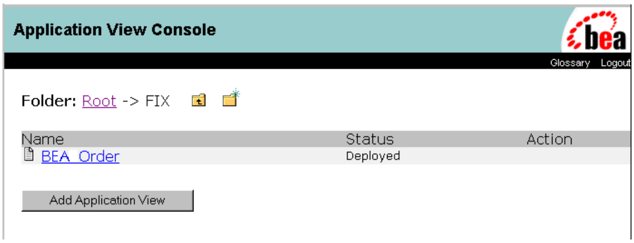
**Figure 3-12 WebLogic Server Log**



```
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="SDValue" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:NMTOKEN">
            <xsd:enumeration value="Filled"/>
            <xsd:enumeration value="New"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:complexType>
  </xsd:element>

  <xsd:element name="LeavesQty" type="xsd:short"/>

  <xsd:element name="LastPx" type="xsd:decimal"/>

</xsd:schema>

INFO  17 Aug 2002 23:28:43,589 BEA_FIX_1_0.EventGenerator  - event generator com.ibi.bea
fix.event.EventGenerator@d6cb01a started
INFO  17 Aug 2002 23:28:43,589 BEA_FIX_1_0.EventGenerator  - event generator com.ibi.bea
fix.event.EventGenerator@d6cb01a refreshed
DEBUG 17 Aug 2002 23:28:46,883 BEA_FIX_1_0.DesignTime 9e25QAPXbkfyE0zUlBv0jAM71SKrIruYzu
G4Xg3Acxt1AgrBDru8!260797988!-1062731420!7001!7002!1029640520244:system[platform=Windows
, browser=MS Internet Explorer, version=5.5] - controller >> redirected to /wlai/display
.jsp?content=appvwsum&template=adapter&qualifiedAppViewName=FIX.IBM_Order
INFO  17 Aug 2002 23:29:39,399 BEA_FIX_1_0.EventGenerator  - INFO [FIX] W.FIX.1: Worker
accepts work
```

10. To validate that the application view is successfully deployed, proceed to the main Application View Console page, and select the folder in which you created the application view. You should see the name of the new application view with a status of deployed, as shown in the following figure:

**Figure 3-13 Application View Console: New Application View**



Application View Console

Folder: Root -> FIX

| Name | Status | Action |
|---|---|---|
| BEA_Order | Deployed | |

You have finished configuring the event adapter application view.

You can confirm that you configured it correctly and that it can successfully receive events. See "Testing Events Using Application View Console" on page 3-12 and "Testing Application View Events Using Studio" on page 3-14.

# Testing Events Using Application View Console

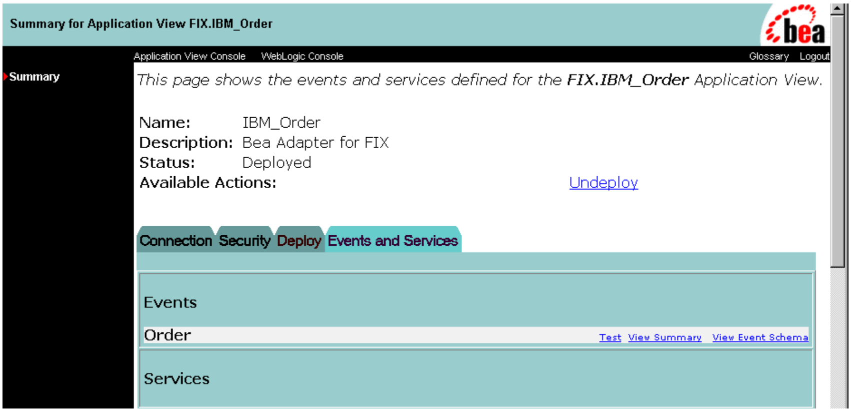
You can use the Application View Console to confirm that a deployed event adapter application view is correctly configured and can receive events. You can also use the WebLogic Integration Studio to test an event. For more information, see "Testing Application View Events Using Studio" on page 3-14.

To test events using the application view console:

1. Log on to the WebLogic Integration Application View Console at //*appserver-host:port*/wlai.

2. Select the folder in which the application view resides, and then select the application view. The Summary page opens:

**Figure 3-14 Application View Console: Summary**



Summary for Application View FIX.IBM_Order

*This page shows the events and services defined for the* ***FIX.IBM_Order*** *Application View.*

**Name:** IBM_Order
**Description:** Bea Adapter for FIX
**Status:** Deployed
**Available Actions:** Undeploy

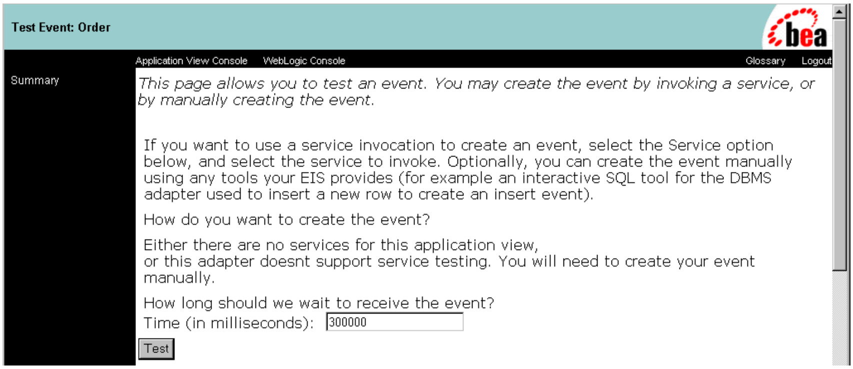
Events

Order — Test View Summary View Event Schema

Services

3. Click Test for one of the application view's events. The Test Event page opens:

**Figure 3-15 Application View Console: Test Event**



Test Event: Order

*This page allows you to test an event. You may create the event by invoking a service, or by manually creating the event.*

If you want to use a service invocation to create an event, select the Service option below, and select the service to invoke. Optionally, you can create the event manually using any tools your EIS provides (for example an interactive SQL tool for the DBMS adapter used to insert a new row to create an insert event).

How do you want to create the event?

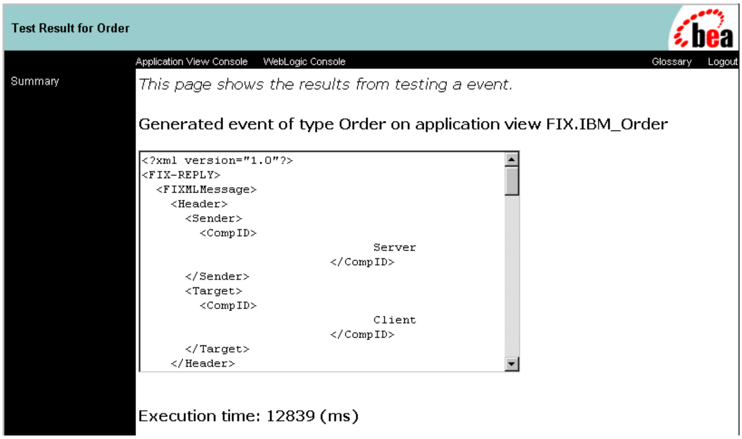
Either there are no services for this application view,
or this adapter doesnt support service testing. You will need to create your event manually.

How long should we wait to receive the event?
Time (in milliseconds): 300000

4. Enter 3000 (or a higher value) in the Time box. This will provide a 30-second period during which, in the following step, you can access the a FIX client program (or your favorite utility) to manually invoke a request from FIX Client to your event adapter.

5. In the Application View Console, the Test Result page will soon display the event's result:

**Figure 3-16 Application View Console: Test Result**



*This page shows the results from testing a event.*

Generated event of type Order on application view FIX.IBM_Order

```
<?xml version="1.0"?>
<FIX-REPLY>
  <FIXMLMessage>
    <Header>
      <Sender>
        <CompID>
                                Server
                          </CompID>
      </Sender>
      <Target>
        <CompID>
                                Client
                          </CompID>
      </Target>
    </Header>
```

Execution time: 12839 (ms)

If you wait longer than a minute and do not receive the event's result, you should assume that there is a problem with the event adapter application view. Examine the WebLogic Server Log for information about the event's activity.

Otherwise, you have now confirmed that the event adapter application view is correctly configured and can receive events.

# Testing Application View Events Using Studio

You can use WebLogic Integration Studio to confirm that a deployed event adapter application view is correctly configured and can receive events. You can also test events using the Application View Console. For more information, see "Testing Events Using Application View Console" on page 3-12.

To test application view events using Studio:

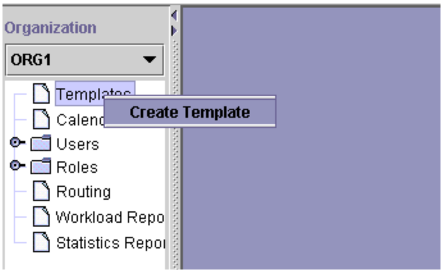
1. Start the WebLogic Integration Studio:

   **On a Windows system,** choose Start→Programs→BEA WebLogic Platform 7.0→WebLogic Integration 7.0→Studio.

**On a UNIX system,** go to the WLI_HOME/bin directory and run the studio command.

2. Log on to the WebLogic Integration Studio.
3. In the Organization panel, choose an organization to create a new Studio workflow template.
4. Right-click Templates and select Create Template:

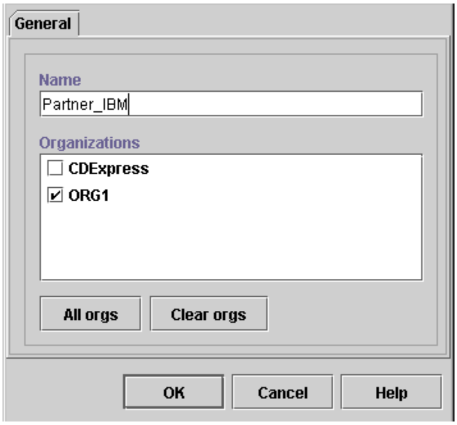
**Figure 3-17 WebLogic Integration Studio: Create Template**



Organization: ORG1

5. Select a name for your Workflow and Click OK.

**Figure 3-18 Template Properties**

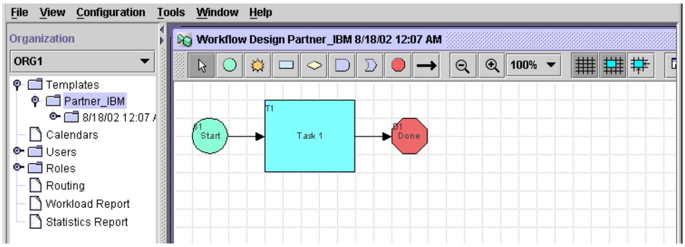


Name: Partner_IBM

Organizations:
☐ CDExpress
☑ ORG1

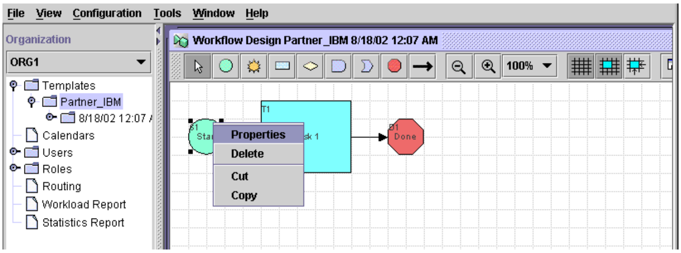
6. Right Click the new template and select Create Template Definition. The template is displayed in WebLogic Integration Studio:

**Figure 3-19 WebLogic Integration Studio: New Template**



7. Right-click the Start node and select Properties:

**Figure 3-20 WebLogic Integration Studio: Start Node Properties**



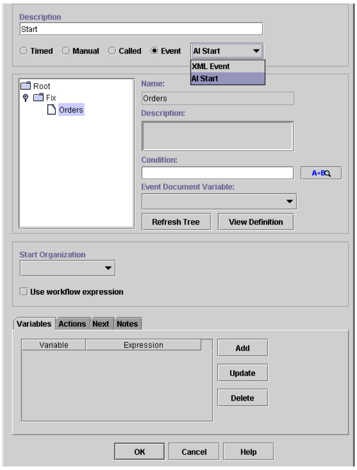
The Start Properties dialog box appears.

8. Select Event→AI Start.

9. In the event explorer, browse the application view folders and select the application view that corresponds to the event adapter.

10. Open the event adapter and select the desired event:

**Figure 3-21 WebLogic Integration Studio: Start Properties**



Description: Start

○ Timed ○ Manual ○ Called ● Event — AI Start

Name: Orders

Description:

Condition:

Event Document Variable:

Start Organization:

□ Use workflow expression

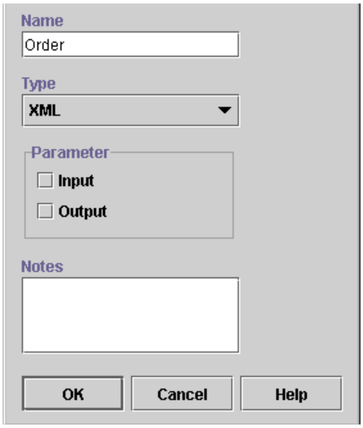
| Variable | Expression |
|---|---|

11. Select <new> from the Event Document Variable drop-down list box:

    The Variable Properties dialog box appears.

12. Type a name for the new variable, select the variable type XML, and check the Input and Output options in the Parameter group:

**Figure 3-22 WebLogic Integration Studio: Variable Properties**



Name: Order

Type: XML

Parameter

□ Input

□ Output

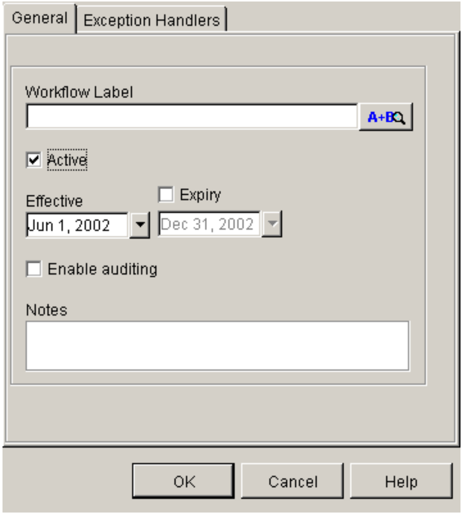
Notes:

13. Click OK.

14. Right-click the template in WebLogic Integration Studio's left panel and select Save.

15. Right-click the event definition folder and choose Properties. The Template Definition dialog box appears.

**Figure 3-23 WebLogic Integration Studio: Template Definition**



Workflow Label

☑ Active

Effective: Jun 1, 2002

□ Expiry: Dec 31, 2002

□ Enable auditing

Notes

16. Ensure that Active is checked, and click OK.

17. You may now initiate events from your Enterprise Information System. For the BEA WebLogic Adapter for FIX, you can create events through a business application or through a FIX Client test program.

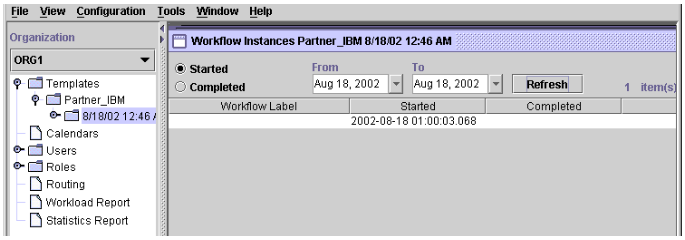
18. Return to WebLogic Integration Studio.

19. Right-click the event definition folder and select Instances:

    The Workflow Instances for your event definition appear. You can now track execution of your workflow.

20. Select Started and click Refresh. You should now see a list of started workflows:

**Figure 3-24 WebLogic Integration Studio: Workflow Instances**



● Started
○ Completed

From: Aug 18, 2002
To: Aug 18, 2002

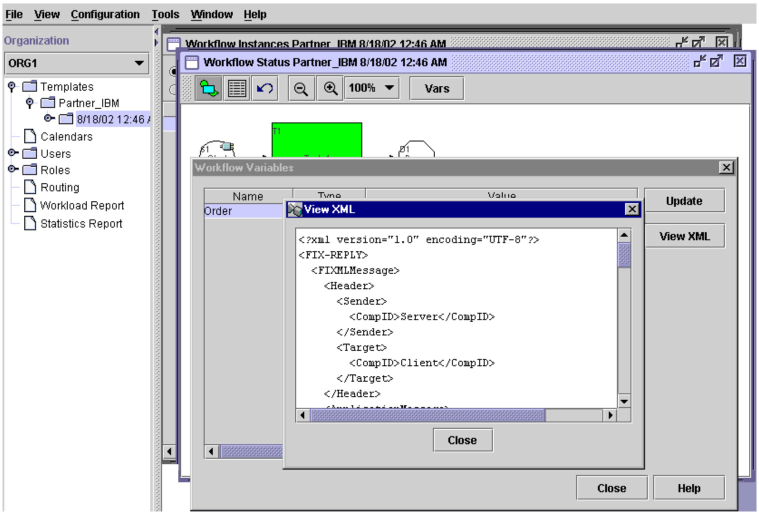
1 item(s)

| Workflow Label | Started | Completed |
|---|---|---|
|  | 2002-08-18 01:00:03.068 |  |

21. Right-click any instance of the workflow and choose Variables. The Workflow Variables dialog box appears.

22. Click View XML to see the entire contents of the workflow message/document:

**Figure 3-25 WebLogic Integration Studio: Workflow Variables**



| Name | Type | Value |
|---|---|---|
| Order |  |  |

```
<?xml version="1.0" encoding="UTF-8"?>
<FIX-REPLY>
  <FIXMLMessage>
    <Header>
      <Sender>
        <CompID>Server</CompID>
      </Sender>
      <Target>
        <CompID>Client</CompID>
      </Target>
    </Header>
```

For more information on using the Studio, see “Using Application Views in the Studio” in *Using Application Integration*:

- For WebLogic Integration 7.0, see `http://edocs.bea.com/wli/docs70/aiuser/3usruse.htm`

- For WebLogic Integration 2.1, see `http://edocs.bea.com/wlintegration/v2_1sp/aiuser/3usruse.htm`

# 4 Creating and Configuring a Service Adapter

The service adapter for FIX is WebLogic Integration's interface to Financial Information eXchange (FIX) protocol. It enables your business processes to communicate with FIX Servers. This section describes how to create, configure, and test a service adapter. It includes the following topics:

- Creating a Service Adapter Application View
- Configuring a Service Adapter Application View
- Testing the Service Adapter for FIX
- Testing Application View Services Using Studio

## Creating a Service Adapter Application View

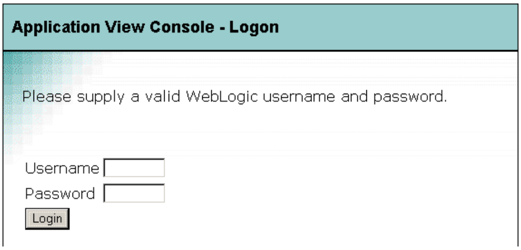
The service adapter for FIX will be configured for all services that need to send data to FIX Servers.

To create a service adapter application view:

1. Log on to the WebLogic Integration Application View Console at //*appserver-host:port*/wlai, where *appserver-host* is the IP address or host name where the WebLogic Integration Server is installed, and *port* is the socket on which the server is listening. The port, if not changed during installation, defaults to 7001.

2. If prompted, enter a user name and password, as shown in the following figure:

**Figure 4-1 Application View Console Logon Window**



Application View Console - Logon

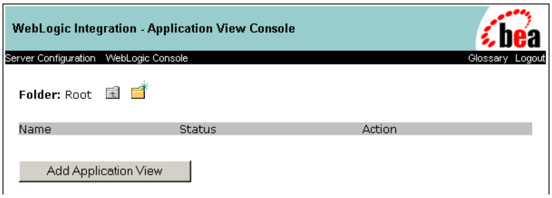
Please supply a valid WebLogic username and password.

**Note:** If the user name is not `system`, it must be included in the `adapter` group. For more information on adding the administrative server user name to the `adapter` group, see the *BEA WebLogic Adapter for FIX Installation and Configuration Guide*.

3. Click Login.

   The WebLogic Integration Application View Console opens.

**Figure 4-2 Application View Console Window**



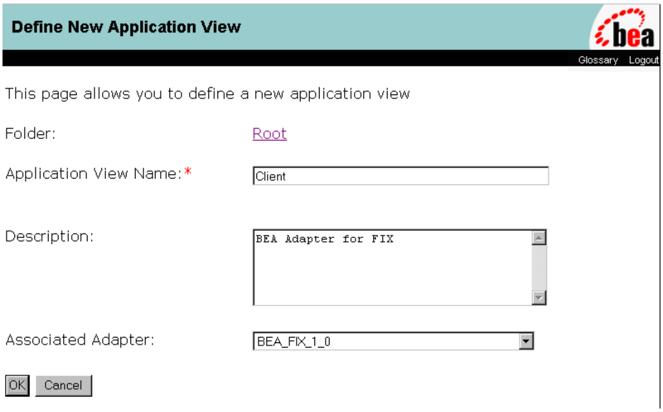
WebLogic Integration - Application View Console

4. Click Add Application View. The Define New Application View page opens. An application view enables a set of business processes for this adapter's target EIS application. For more information, see “Defining an Application View” in *Using Application Integration*:

   - For WebLogic Integration 7.0, see `http://edocs.bea.com/wli/docs70/aiuser/2usrdef.htm`

- For WebLogic Integration 2.1, see `http://edocs.bea.com/wlintegration/v2_1sp/aiuser/2usrdef.htm`

**Figure 4-3 Define New Application View**



Define New Application View

This page allows you to define a new application view

Folder: Root

Application View Name:* Client

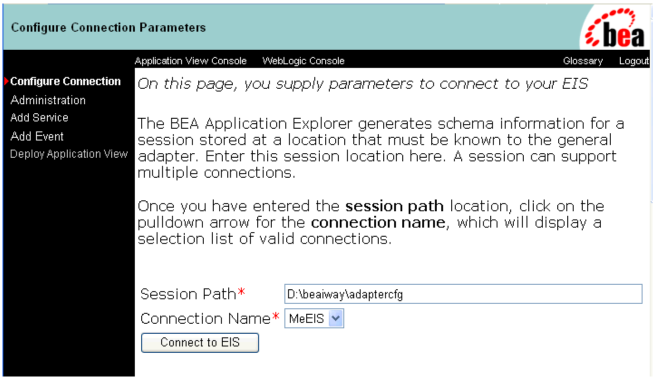
Description: BEA Adapter for FIX

Associated Adapter: BEA_FIX_1_0

5. Give the Application View a name and description, and select BEA_FIX_1_0 from the Associated Adapter list, as illustrated in the following figure.

6. Click OK. The Configure Connection Parameters page opens.

**Figure 4-4 Configure Connection Parameters**



Configure Connection Parameters

*On this page, you supply parameters to connect to your EIS*

The BEA Application Explorer generates schema information for a session stored at a location that must be known to the general adapter. Enter this session location here. A session can support multiple connections.

Once you have entered the **session path** location, click on the pulldown arrow for the **connection name**, which will display a selection list of valid connections.

Session Path* D:\beaiway\adaptercfg

Connection Name* MeEIS

7. Supply the name of the BEA WebLogic Adapter for FIX session base directory in the Session Path box. This directory holds your FIX schema information, and contains the subdirectory FIX/*YourConnectionName*.

For example, the session base directory might be `d:\bea\bse\sessions\default`, with the schema repository—containing a repository manifest and schemas—residing in `d:\bea\bse\sessions\default\FIX\FIX4.2`.

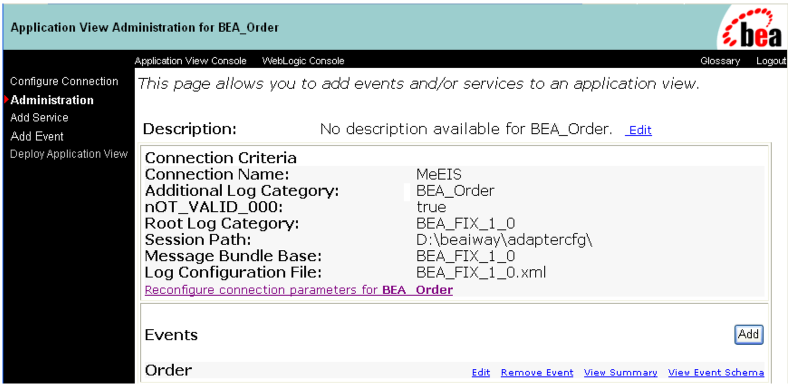
8. Select the session name—also known as the connection name—from the Connection name drop-down list box.
9. Click Connect to EIS. The Application View Administration page opens.

   Note that you can access the Configure Connection Parameters page (displayed in the previous step) when the Application View is not deployed, simply by selecting the Reconfigure connection parameters link. If the Application View is deployed, you can access the page by first undeploying the Application View.

# Configuring a Service Adapter Application View

After you create an application view, you can configure a service for the Application View. For more information on creating an application view, see "Creating a Service Adapter Application View" on page 4-1.

**Figure 4-5 Application View Administration Window**



*This page allows you to add events and/or services to an application view.*

Description: No description available for BEA_Order.

Connection Criteria

| | |
|---|---|
| Connection Name: | MeEIS |
| Additional Log Category: | BEA_Order |
| nOT_VALID_000: | true |
| Root Log Category: | BEA_FIX_1_0 |
| Session Path: | D:\beaiway\adaptercfg\ |
| Message Bundle Base: | BEA_FIX_1_0 |
| Log Configuration File: | BEA_FIX_1_0.xml |

Events

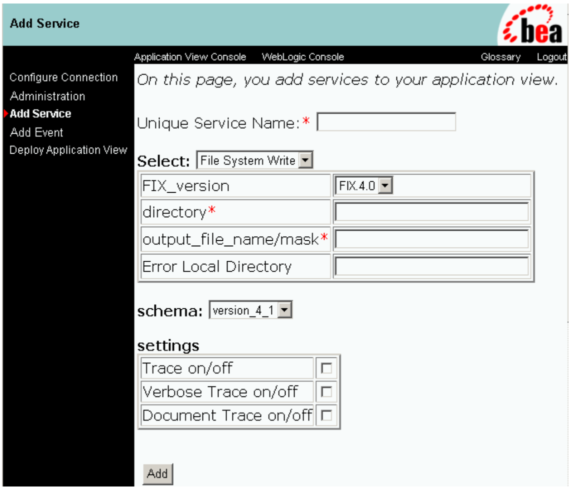
Order

To configure a service adapter application view:

1. If it is not already open, open the application view to be modified. For more information, see “Editing an Application View” in “Defining an Application View” in *Using Application Integration*:
   - For WebLogic Integration 7.0, see `http://edocs.bea.com/wli/docs70/aiuser/2usrdef.htm`
   - For WebLogic Integration 2.1, see `http://edocs.bea.com/wlintegration/v2_1sp/aiuser/2usrdef.htm`
2. From the left navigation panel of the Application View Administration page, select Add Service. The Add Service page opens.

   Two protocols are available: File and FTP.

**Figure 4-6 Add Service - File System Write**



On this page, you add services to your application view.

Unique Service Name:*

Select: File System Write

| | |
|---|---|
| FIX_version | FIX 4.0 |
| directory* | |
| output_file_name/mask* | |
| Error Local Directory | |

schema: version_4_1

settings

| | |
|---|---|
| Trace on/off | □ |
| Verbose Trace on/off | □ |
| Document Trace on/off | □ |

The following table describes the properties in the File System Write window.

**Table 4-1 Add Service - File System Write Properties**

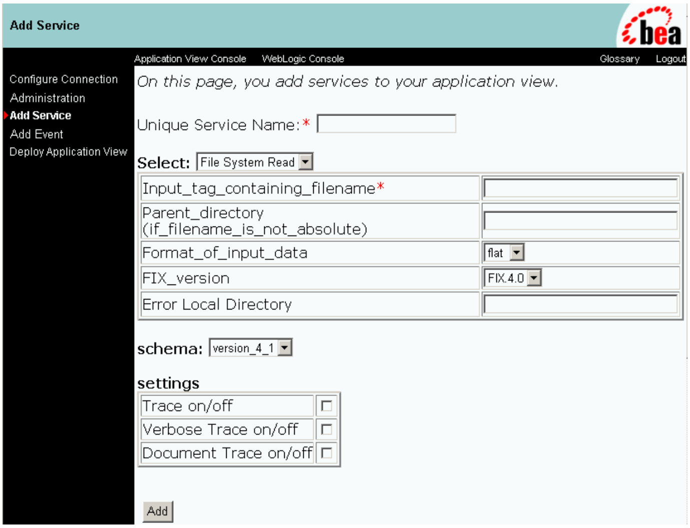
| Property | Description | Type | Sample Value |
|---|---|---|---|
| FIX_version | The version of FIX. | drop down | 4_1, 4_2, 4_3 |
| directory (required) | Directory to which output messages are emitted. | string | |

**Table 4-1 Add Service - File System Write Properties**

| Property | Description | Type | Sample Value |
|---|---|---|---|
| output_file_name/ mask (required) | The output file name (can contain a '*'), which gets expanded to a timestamp.<br>A pound symbol can be used as a mask for a sequence count. Each pound symbol represents a whole number integer value. For example, File## counts up to 99 before restarting at 0, File### counts up to 999 before restarting at 0, and so on. | string | |
| Error Local Directory | Directory to which the document being processed is written to in the event of a failure (for instance if the transformation fails or the document does not match the schema). | directory path | |

The following window shows the parameters for the File System Read protocol.

**Figure 4-7 Add Service - File System Read**



*On this page, you add services to your application view.*

Unique Service Name:*

Select: File System Read

| | |
|---|---|
| Input_tag_containing_filename* | |
| Parent_directory (if_filename_is_not_absolute) | |
| Format_of_input_data | flat |
| FIX_version | FIX.4.0 |
| Error Local Directory | |

schema: version_4_1

settings

| | |
|---|---|
| Trace on/off | □ |
| Verbose Trace on/off | □ |
| Document Trace on/off | □ |

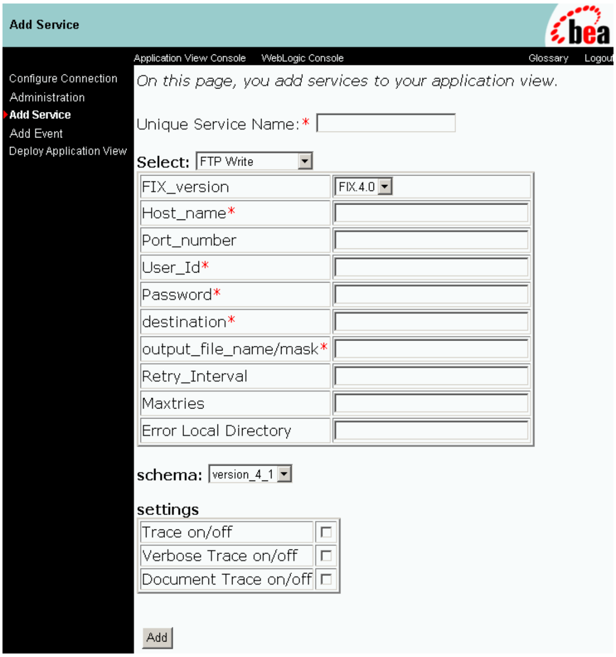
The following table describes the properties in the File System Read window.

**Table 4-2 Add Service - File System Read Properties**

| Property | Description | Type | Sample Value |
|---|---|---|---|
| Input_tag_containing_filename (required) | Name of the XML tag that contains the name of the file in the file system. The file can be XML or non-XML format and can be transformed as part of the configuration. | string | |
| Parent_directory (if_filename_is_not_absolute) | Optional path to the file, if it is not part of the Input tag. | string | |
| Format_of_input_data | Format of the file being read.<br>XML format or flat format for non-XML output. | drop down | |
| FIX_version | The version of FIX. | drop down | 4_1, 4_2, 4_3 |
| Error Local Directory | Directory to which the document being processed is written to in the event of a failure (for instance if the transformation fails or the document does not match the schema). | directory path | |

The following window shows the applicable parameters for the FTP Write protocol.

**Figure 4-8 Add Service - FTP Write**



*On this page, you add services to your application view.*

Unique Service Name:*

Select: FTP Write

| | |
|---|---|
| FIX_version | FIX 4.0 |
| Host_name* | |
| Port_number | |
| User_Id* | |
| Password* | |
| destination* | |
| output_file_name/mask* | |
| Retry_Interval | |
| Maxtries | |
| Error Local Directory | |

schema: version_4_1

settings

| | |
|---|---|
| Trace on/off | □ |
| Verbose Trace on/off | □ |
| Document Trace on/off | □ |

The following table describes the properties in the FTP Write window.

**Table 4-3 Add Service - FTP Write Properties**

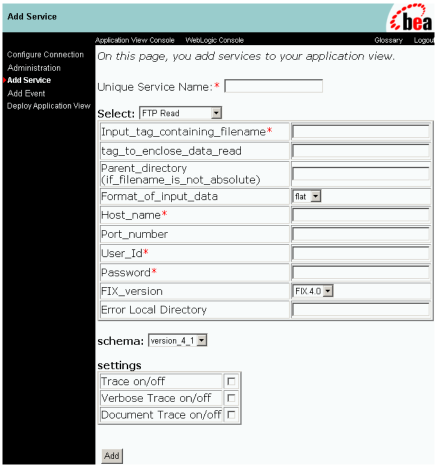
| Property | Description | Type | Sample Value |
|---|---|---|---|
| FIX_version | The version of FIX. | drop down | 4_1, 4_2, 4_3 |
| Host_name (required) | Name of host the machine where the listener will contact the service to obtain requests. | string | |
| Port_number | Port number of the host machine. | string | |
| User_Id (required) | User account to use when connecting to the protocol host. | string | |
| Password (required) | Password for the user account to use when connecting to the protocol host. | string | |

**Table 4-3 Add Service - FTP Write Properties**

| Property | Description | Type | Sample Value |
|---|---|---|---|
| destination (required) | Directory to address on the FTP target system. | string | |
| output_file_name/ mask | The output file name (can contain a '*'), which gets expanded to a timestamp. | string | |
| Retry_Interval | The maximum wait interval between retries when a connection fails.<br>Retry interval duration in *xx*H:*xx*M:*xx*S format. (for example, 1H:2M:3S, which is 1 hour 2 minutes and 3 seconds) | integer | |
| Maxtries | Number of retries for a failed attempt to write. | integer | |
| Error Local Directory | Directory to which the document being processed is written to in the event of a failure (for instance if the transformation fails or the document does not match the schema). | directory path | |

The following window shows the parameters for the FTP Read protocol.

**Figure 4-9 Add Service - FTP Read**



*On this page, you add services to your application view.*

Unique Service Name:*

Select: FTP Read

| | |
|---|---|
| Input_tag_containing_filename* | |
| tag_to_enclose_data_read | |
| Parent_directory (if_filename_is_not_absolute) | |
| Format_of_input_data | flat |
| Host_name* | |
| Port_number | |
| User_Id* | |
| Password* | |
| FIX_version | FIX.4.0 |
| Error Local Directory | |

schema: version_4_1

settings

| | |
|---|---|
| Trace on/off | □ |
| Verbose Trace on/off | □ |
| Document Trace on/off | □ |

The following table describes the properties in the FTP Read window.

**Table 4-4 Add Service - FTP Read Properties**

| Property | Description | Type | Sample Value |
|---|---|---|---|
| Input_tag_containing_filename (required) | Name of the XML tag that contains the name of the file in the file system. The file can be XML or non-XML format and can be transformed as part of the configuration. | string | |
| tag_to_enclose_data_read | Not required. | string | |
| Parent_directory (if_filename_is_not_absolute) | Optional path to the file, if it is not part of the Input tag. | string | |
| Format_of_input_data | Use the default value of "flat." | drop down | |
| Host_name (required) | Name of host the machine where the listener will contact the service to obtain requests. | string | |
| Port_number | Port number of the host machine. | string | |
| User_Id (required) | User account to use when connecting to the protocol host. | string | |
| Password (required) | Password for the user account to use when connecting to the protocol host. | string | |
| FIX_version | The version of FIX. | drop down | 4_1, 4_2, 4_3 |
| Error Local Directory | Directory to which the document being processed is written to in the event of a failure (for instance if the transformation fails or the document does not match the schema). | directory path | |

3. Select the manifest that describes the schema from the drop-down list.

4. Select the appropriate trace settings.

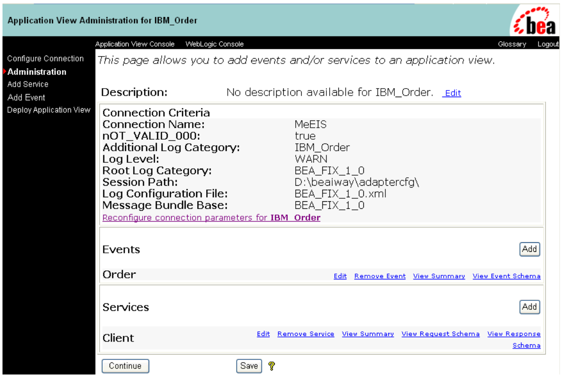
The following table lists and describes each trace setting:

**Table 4-5 Trace Setting Parameters**

| Setting | Definition |
|---|---|
| Trace on/off | Basic traces. Displays the input XML (up to 300 bytes) before parsing, and shows the request being processed. The default setting is off For more information about tracing, see Chapter 5, “Using Tracing.” |
| Verbose Trace on/off | More extensive traces. Displays configuration parameters used by the adapter. The default setting is off. For more information about tracing, see Chapter 5, “Using Tracing.” |
| Document Trace on/off | Displays the input document after it was analyzed and the response document being returned. Because some documents are very large, this trace category can severely affect performance and memory use. The default setting is off. For more information about tracing, see Chapter 5, “Using Tracing.” |

5. Click Add. The Application View Administration window appears, as shown in the following figure.

**Figure 4-10 Application View Administration Window**



Application View Administration for IBM_Order

*This page allows you to add events and/or services to an application view.*

Description: No description available for IBM_Order. Edit

Connection Criteria

| | |
|---|---|
| Connection Name: | MeEIS |
| nOT_VALID_000: | true |
| Additional Log Category: | IBM_Order |
| Log Level: | WARN |
| Root Log Category: | BEA_FIX_1_0 |
| Session Path: | D:\beaiway\adaptercfg\ |
| Log Configuration File: | BEA_FIX_1_0.xml |
| Message Bundle Base: | BEA_FIX_1_0 |

Reconfigure connection parameters for **IBM_Order**

Events

Order

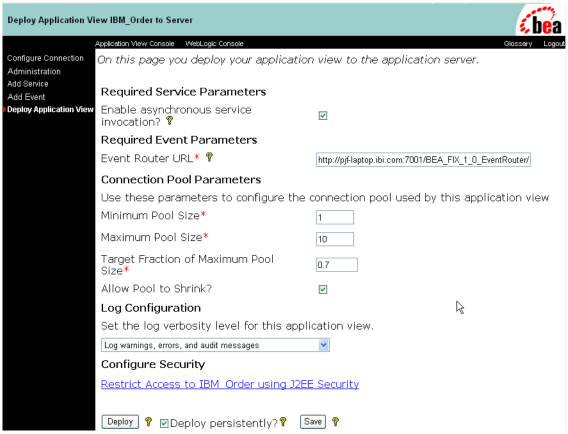
Services

Client

6. Click Continue.

   The Deploy Application View window appears.

**Figure 4-11 Deploy Application View**



*On this page you deploy your application view to the application server.*

Required Service Parameters

Enable asynchronous service invocation? ☑

Required Event Parameters

Event Router URL* http://pjf-laptop.ibi.com:7001/BEA_FIX_1_0_EventRouter/

Connection Pool Parameters

Use these parameters to configure the connection pool used by this application view

Minimum Pool Size* 1

Maximum Pool Size* 10

Target Fraction of Maximum Pool Size* 0.7

Allow Pool to Shrink? ☑

Log Configuration

Set the log verbosity level for this application view.

Log warnings, errors, and audit messages

Configure Security

Restrict Access to IBM_Order using J2EE Security

☑ Deploy persistently?

7. Make any changes to the Deploy Application View page and click Deploy.

   The Summary for Application View Client page displays on successful completion of the Application View Deploy. You can confirm that you configured it correctly and that it can successfully receive events using two methods: by using the Application View Console or by using the WebLogic Integration Studio. For more information, see "Testing the Service Adapter for FIX" on page 4-12and "Testing Application View Services Using Studio" on page 4-16.

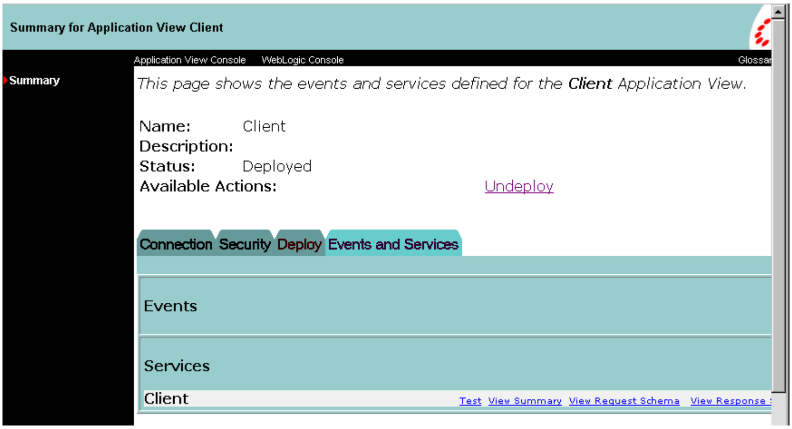
# Testing the Service Adapter for FIX

The service adapter connects to a FIX Server and returns a status.

To test the service adapter for FIX:

1. From the Summary for Application View page, click Test.

### Figure 4-12 Test the Service



**Summary for Application View Client**

*This page shows the events and services defined for the **Client** Application View.*

Name: Client
Description:
Status: Deployed
Available Actions: Undeploy

Events

Services

Client Test View Summary View Request Schema View Response S

2. Enter a sample document that matches the request schema for the configured service. The class XDFIXpreParser is used to convert the FIX inbound message into XML. The outbound FIX messages are sent in FIX format (Tag based and not XML based). For example, the OrderIn request schema has an instance document like the following:

### Listing 4-1 Order Document

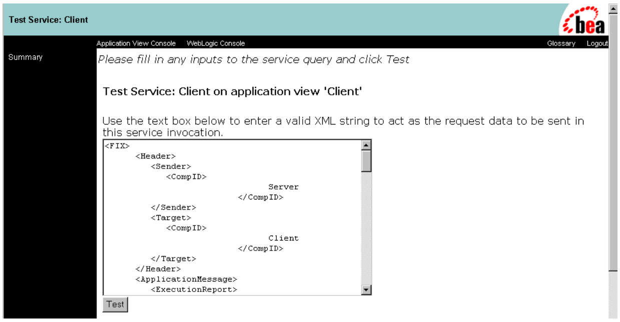
```
<FIX>
      <Header>
         <Sender>
            <CompID>
                            Server
                       </CompID>
         </Sender>
         <Target>
            <CompID>
                            Client
                       </CompID>
         </Target>
      </Header>
      <ApplicationMessage>
         <ExecutionReport>
           <OrderID>
                            iWay
                      001</OrderID>
            <ClOrdID>
```

```
                         iWay
                    </ClOrdID>
         <ExecID>1234</ExecID>
         <ExecTransType>
            <ExecNew>
               <LastShares>1000</LastShares>
               <LastPx>80.01</LastPx>
            </ExecNew>
         </ExecTransType>
         <ExecType Value="0" SDValue="New"/>
         <OrderStatus Value="0" SDValue="New"/>
         <Instrument>
            <Symbol>
                                IBM
                           </Symbol>
         </Instrument>
         <Side Value="1" SDValue="Buy"/>
         <OrderQty>
                               100
                           </OrderQty>
         <LeavesQty>222</LeavesQty>
         <CumQty>100</CumQty>
         <AvgPx>123.45</AvgPx>
      </ExecutionReport>
   </ApplicationMessage>
</FIXMLMessage>
<FIXMLMessage>
   <Header>

      <Sender>
         <CompID>
                         Server
                    </CompID>
      </Sender>
      <Target>
         <CompID>
                         Client
         </CompID>
      </Target>
   </Header>
   <ApplicationMessage>
      <ExecutionReport>
         <OrderID>
                         iWay
                    001</OrderID>
         <ClOrdID>
                         iWay
                    </ClOrdID>
         <ExecID>1234</ExecID>
```

```
            <ExecTransType>
               <ExecNew>
                  <LastShares>1000</LastShares>
                  <LastPx>80.01</LastPx>
               </ExecNew>
            </ExecTransType>
            <ExecType Value="0" SDValue="New"/>
            <OrderStatus Value="2" SDValue="Filled"/>
            <Instrument>
               <Symbol>
                                   IBM
                              </Symbol>
            </Instrument>
            <Side Value="1" SDValue="Buy"/>
            <OrderQty>
                                   100
                              </OrderQty>
            <LeavesQty>222</LeavesQty>
            <CumQty>100</CumQty>
            <AvgPx>123.45</AvgPx>
         </ExecutionReport>
      </ApplicationMessage>
   </FIXMLMessage>
</FIX>
```

3. Enter this document into the Service Test page by either typing or by copying and pasting into the page.

**Figure 4-13 Test Service: OrderIn**



*Please fill in any inputs to the service query and click Test*

Test Service: Client on application view 'Client'

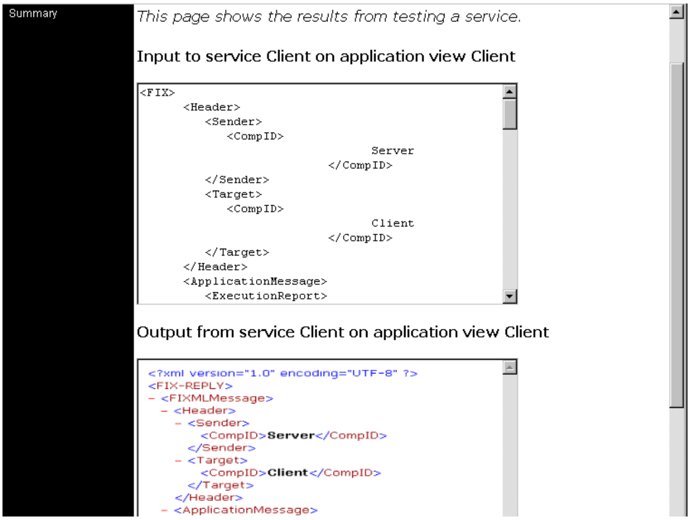
Use the text box below to enter a valid XML string to act as the request data to be sent in this service invocation.

```
<FIX>
      <Header>
         <Sender>
            <CompID>
                                Server
                          </CompID>
         </Sender>
         <Target>
            <CompID>
                                Client
                          </CompID>
         </Target>
      </Header>
      <ApplicationMessage>
         <ExecutionReport>
```

4. Select Test to send the request to a FIX Server.

The response document should indicate the success of emitting FIX message. See the sample response below:

**Figure 4-14 Sample Response**



*This page shows the results from testing a service.*

Input to service Client on application view Client

```
<FIX>
      <Header>
         <Sender>
            <CompID>
                              Server
                        </CompID>
         </Sender>
         <Target>
            <CompID>
                              Client
                        </CompID>
         </Target>
      </Header>
      <ApplicationMessage>
         <ExecutionReport>
```

Output from service Client on application view Client

```
<?xml version="1.0" encoding="UTF-8" ?>
<FIX-REPLY>
- <FIXMLMessage>
  - <Header>
    - <Sender>
        <CompID>Server</CompID>
      </Sender>
    - <Target>
        <CompID>Client</CompID>
      </Target>
    </Header>
  - <ApplicationMessage>
```

5. After any Client logic a conversation will be initiated with the FIX server and a response will be returned to the Service.

# Testing Application View Services Using Studio

You can use WebLogic Integration Studio to confirm that a deployed event adapter application view is correctly configured.

To test application view services using Studio:

1. Start the WebLogic Integration Studio:

   **On a Windows system,** choose Start→Programs→BEA WebLogic Platform 7.0→WebLogic Integration 7.0→Studio.

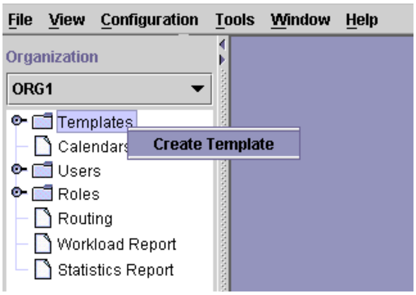
**On a UNIX system,** go to the WLI_HOME/bin directory and run the studio command.

2. Log on to the WebLogic Integration Studio.

3. In the Organization panel, choose an organization to create a new Studio workflow template.

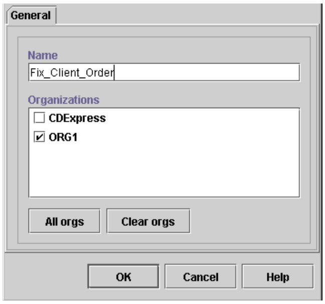
4. Right-click Templates and select Create Template:

**Figure 4-15 WebLogic Integration Studio: Create Template**



5. Select an name for your Workflow and Click OK.

**Figure 4-16 Template Properties**



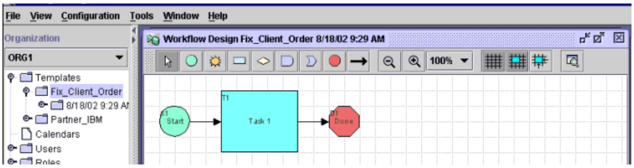
Name
Fix_Client_Order

Organizations
☐ CDExpress
☑ ORG1

6. Right Click the new template and select Create Template Definition.

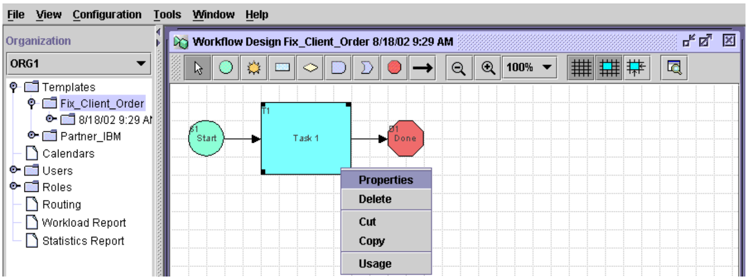
   The template is displayed in WebLogic Integration Studio:

**Figure 4-17 WebLogic Integration Studio: New Template**



7. Right-click the Task node and select Properties:

**Figure 4-18 WebLogic Integration Studio: Start Node Properties**



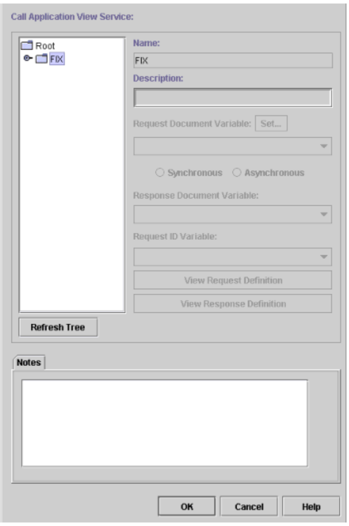
The Start Properties dialog box appears.

8. Select ADD→AI Actions→Call Application View Service.

9. In the Service explorer, browse the Application View folders and select the application view that corresponds to the service adapter.

10. Open the service adapter and select the desired service:

**Figure 4-19 WebLogic Integration Studio: Call Service**

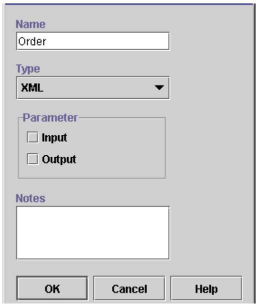


Name: FIX

11. Select <new> from the Request Document Variable drop-down list box. The Variable Properties dialog box opens

12. Type a name for the new variable, select the variable type XML, and check the Input and Output options in the Parameter group:

**Figure 4-20 WebLogic Integration Studio: Variable Properties**



Name: Order

Type: XML

13. Click OK.

14. Select Synchronous on the Call Service Dialog.

15. Select <new> from the Response Document Variable drop-down list box:

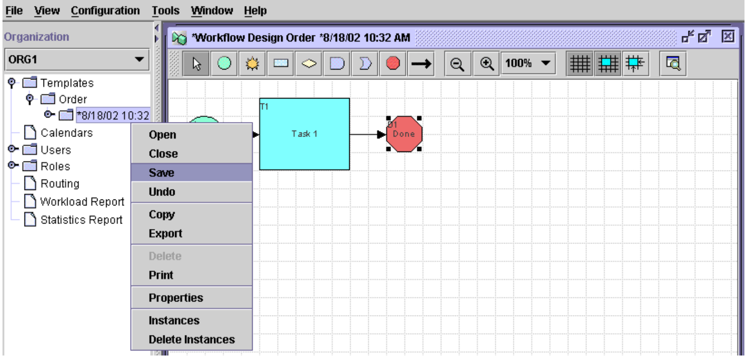
16. Type a name for the new variable, select the variable type XML.

17. Since this is only a partial workflow, the request document containing the request must be set. Click set on the Call Service panel to set the request document. This will allow you to choose an XML document containing the service request.

18. Click OK to return the template.

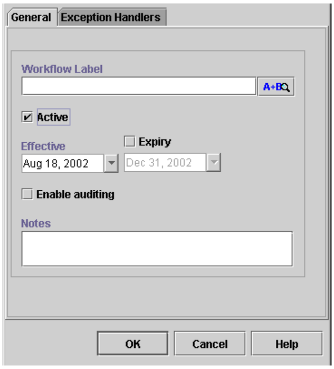
19. Right-click the template in WebLogic Integration Studio's left panel and select Save:

**Figure 4-21 WebLogic Integration Studio: Save Template**



20. Right-click the event definition folder and choose Properties. The Template Definition dialog box appears.

**Figure 4-22 WebLogic Integration Studio: Template Definition**



Workflow Label

☑ Active

Effective: Aug 18, 2002

☐ Expiry: Dec 31, 2002

☐ Enable auditing

Notes

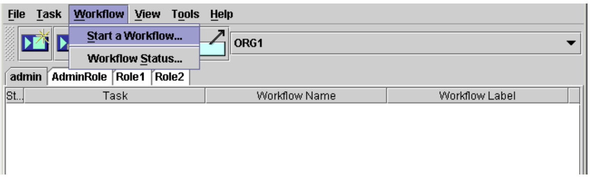
21. Ensure that Active is checked, and click OK.

22. You may now initiate the workflow from the BEA Wordlist. Select BEA WebLogic E-Business Platform→WebLogic Integration→Wordlist

23. Logon on to Wordlist.

24. Select Workflow Start.

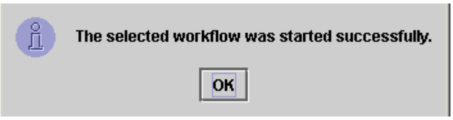
**Figure 4-23 WebLogic Integration Worklist**



25. Select the workflow that was just created from the workflow list and click OK.

26. You will receive a message indicating that your workflow has been started successfully.

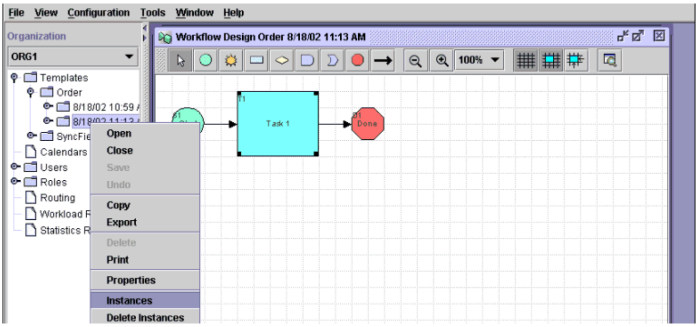
**Figure 4-24 Success Message**


The selected workflow was started successfully.

The Workflow Instances for your service definition appear. You can now track execution of your workflow.

27. Right-click the service definition folder and select Instances:

**Figure 4-25 Workflow Instances**

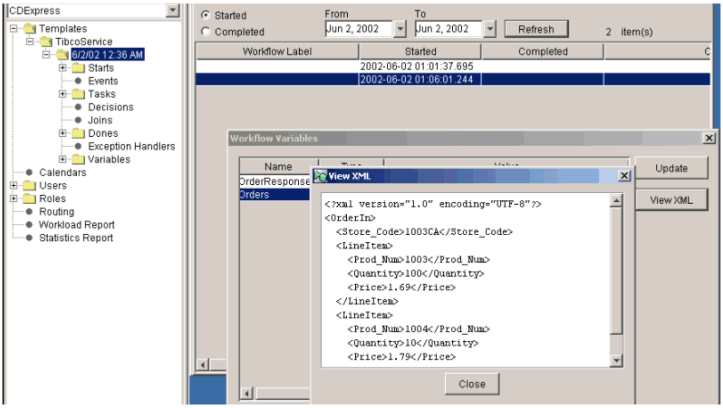


28. Select Started and click Refresh. You should now see a list of workflows.

29. Right-click any instance of the workflow and choose Variables. The Workflow Variables dialog box appears.

30. Click View XML to see the entire contents of the workflow message/document:

**Figure 4-26 WebLogic Integration Studio: Workflow Variables**



```
<?xml version="1.0" encoding="UTF-8"?>
<OrderIn>
  <Store_Code>1003CA</Store_Code>
  <LineItem>
    <Prod_Num>1003</Prod_Num>
    <Quantity>100</Quantity>
    <Price>1.69</Price>
  </LineItem>
  <LineItem>
    <Prod_Num>1004</Prod_Num>
    <Quantity>10</Quantity>
    <Price>1.79</Price>
```

For more information on the Studio, see “Using Application Views in the Studio” in *Using Application Integration*:

- For WebLogic Integration 7.0, see `http://edocs.bea.com/wli/docs70/aiuser/3usruse.htm`
- For WebLogic Integration 2.1, see `http://edocs.bea.com/wlintegration/v2_1sp/aiuser/3usruse.htm`

# 5 Using Tracing

Tracing is an essential feature of an adapter. Most adapters integrate different applications and do not interact with end users while processing data. Unlike a front-end component, when an adapter encounters an error or a warning condition, the adapter cannot stop processing and wait for an end user to respond.

Moreover, many business applications that are connected by adapters are mission-critical. For example, an adapter might maintain an audit report of every transaction with an EIS. Consequently, adapter components must provide both accurate logging and auditing information. The adapter tracing and  logging framework is designed to accommodate both logging and auditing.

This section describes tracing for services and events. It contains the following topics:

- Levels and Categories of Tracing
- Tracing and Performance
- Creating Traces for Services and Events

# Levels and Categories of Tracing

Tracing is provided by both the BEA adapter framework and by the BEA WebLogic Adapter for FIX. The BEA WebLogic Integration framework provides five distinct levels of tracing:

**Table 5-1 Levels of Tracing**

| Level | Indicates |
|---|---|
| AUDIT | An extremely important log message related to the business processing performed by an adapter. Messages with this priority are always written to the log. |
| ERROR | An error in the adapter. Error messages are internationalized and localized for the user. |
| WARN | A situation that is not an error, but that could cause problems in the adapter. Warning messages are internationalized and localized for the user. |
| INFO | An informational message that is internationalized and localized for the user. |
| DEBUG | A debug message, that is, information used to determine how the internals of a component are working. Debug messages usually are not internationalized. |

The adapter framework provides three specialized categories of tracing:

**Table 5-2 Categories of Tracing**

| Level | Indicates |
|---|---|
| Basic Trace | Basic traces. Displays the input XML (up to 300 bytes) before parsing, and shows the request being processed. The default setting is off. |
| Verbose Trace | More extensive traces. Displays configuration parameters used by the adapter. The default setting is off. |

**Table 5-2 Categories of Tracing**

| Level | Indicates |
|---|---|
| Document Trace | Displays the input document after it was analyzed and the response document being returned. Because some documents are very large, this trace category can severely affect performance and memory use. The default setting is off. |

**Note:** To obtain the appropriate trace, both the level and the category must be declared. In a debug situation, BEA Customer Support will request (minimally) a Basic and a Verbose trace.

# Tracing and Performance

The additional trace capabilities provided by the adapter are not strictly hierarchic; rather they are categorized. These traces are designed to provide debugging help with minimum effect on performance. All internal adapter traces are controlled through the additional tracing settings, and all additional settings route their output to the standard debug setting.

If you configure the adapter for additional settings and do not configure standard trace settings, the traces are generated but never appear in output. This affects performance, as the production of the trace continues even though you receive no benefit of the additional trace information.

# Creating Traces for Services and Events

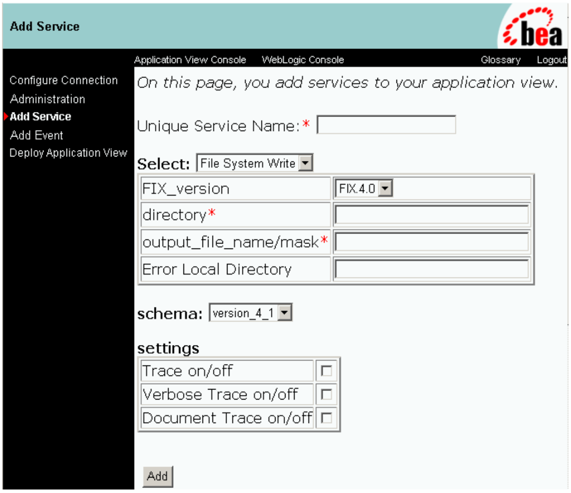
This following topics discuss the steps required to create traces to diagnose adapter problems.

# Creating Traces for a Service

To create traces for a service:

1. Create or modify the service.
2. Ensure that all of the adapter parameters are entered correctly.

**Figure 5-1 Add Service window**



On this page, you add services to your application view.

Unique Service Name:*

Select: File System Write

| | |
|---|---|
| FIX_version | FIX.4.0 |
| directory* | |
| output_file_name/mask* | |
| Error Local Directory | |

schema: version_4_1

settings

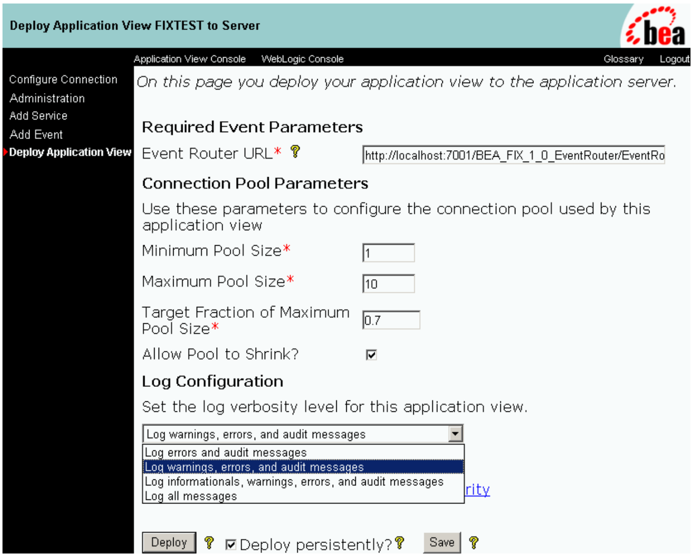
| | |
|---|---|
| Trace on/off | □ |
| Verbose Trace on/off | □ |
| Document Trace on/off | □ |

3. Select the appropriate schema from the drop-down list.
4. Select the appropriate trace levels as described in Table 5-2: Trace, Verbose trace, and Document trace.
5. Click Add to continue to the next configuration pane.
6. Click Continue to move to the next configuration pane.

   The Deploy Application View window opens.
7. Navigate to the Log Configuration area and select the desired trace level.

   This pane enables you to select the trace level for the BEA WebLogic Integration framework.

**Figure 5-2 Deploy Application View window**



*On this page you deploy your application view to the application server.*

Required Event Parameters

Event Router URL* http://localhost:7001/BEA_FIX_1_0_EventRouter/EventRo

Connection Pool Parameters

Use these parameters to configure the connection pool used by this application view

Minimum Pool Size* 1

Maximum Pool Size* 10

Target Fraction of Maximum Pool Size* 0.7

Allow Pool to Shrink? ☑

Log Configuration

Set the log verbosity level for this application view.

Log warnings, errors, and audit messages

- Log errors and audit messages
- Log warnings, errors, and audit messages
- Log informationals, warnings, errors, and audit messages
- Log all messages

rity

☑ Deploy persistently?

For maximum tracing, select Log all Messages.

This is recommended to obtain optimum debugging information for BEA support personnel.

> **Note:** This causes all generated messages to be written to the log. You must select the desired category as defined in Table 5-2 in the adapter to generate the required messages.

8. Click Deploy (or Save) to set the trace settings and deploy the application view.

   Traces are created the next time the service is invoked.

   Traces are output to a file named BEA_FIX_1_0.log in theWebLogic Domain home directory.

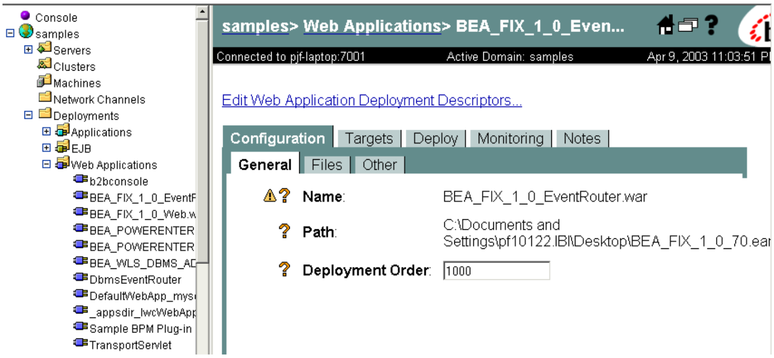
# Creating or Modifying the Tracing Level for an Event

To create or modify the WebLogic framework tracing level for an event:

1. Logon to the BEA WebLogic Server Console.

2. Select Web Applications.
3. Select BEA_FIX_1_0_EventRouter.war.

**Figure 5-3 WebLogic Server Console**



Connected to pjf-laptop:7001 Active Domain: samples Apr 9, 2003 11:03:51 P

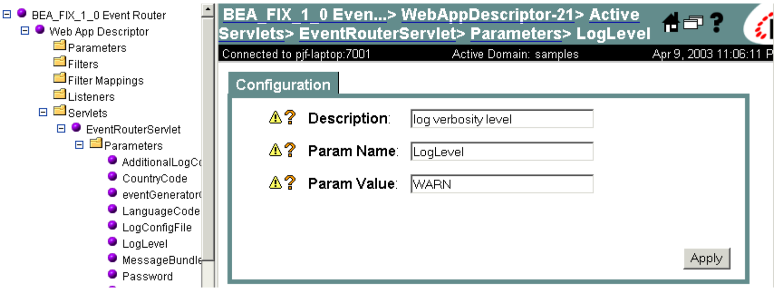
Name: BEA_FIX_1_0_EventRouter.war

Path: C:\Documents and Settings\pf10122.IBI\Desktop\BEA_FIX_1_0_70.ear

Deployment Order: 1000

4. Click Edit Web Application Deployment Descriptors.
5. When the following window opens, select Servlets.
6. In the folder below Servlets, select EventRouterServlet.
7. Select Parameters.
8. Select LogLevel.

**Figure 5-4 WebLogic Server Console: Configuration**



Connected to pjf-laptop:7001 Active Domain: samples Apr 9, 2003 11:06:11 P

Description: log verbosity level

Param Name: LogLevel

Param Value: WARN

This pane enables you to select the trace level for the BEA WebLogic Integration framework.

For maximum tracing, enter DEBUG. This is recommended to obtain optimum debugging information for BEA support personnel

The following levels are valid:

**Table 5-3 Tracing Levels**

| Level | Indicates |
|---|---|
| AUDIT | An extremely important log message related to the business processing performed by an adapter. Messages with this priority are always written to the log. |
| ERROR | An error in the adapter. Error messages are internationalized and localized for the user. |
| WARN | A situation that is not an error, but that could cause problems in the adapter. Warning messages are internationalized and localized for the user. |
| INFO | An informational message that is internationalized and localized for the user. |
| DEBUG | A debug message, that is, information used to determine how the internals of a component are working. Debug messages usually are not internationalized. |

9. Click Apply to save the newly entered trace level.
10. Click BEA_FIX_1_0 EventRouter.
11. Click Persist to apply the logging changes.

    This change need only be made once.

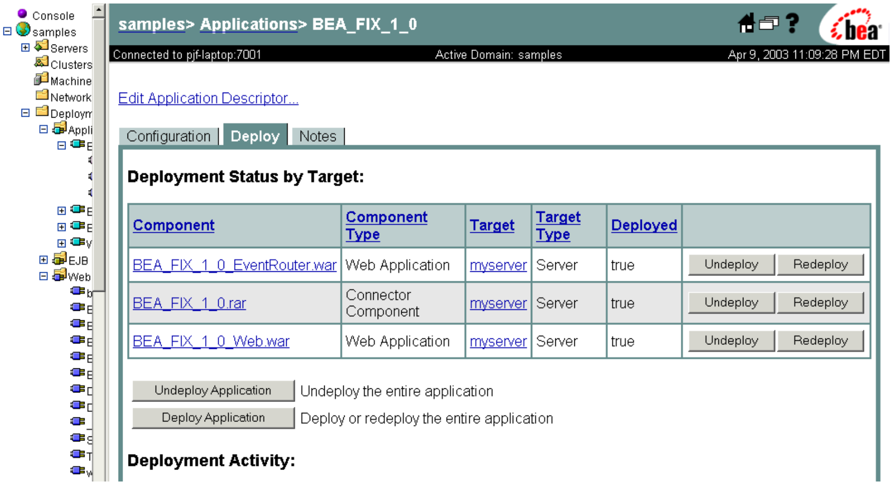
    It is set for all events associated with a given adapter.
12. Return to the WebLogic Server Console.
13. Select Applications from the WebLogic Server Console.
14. Select the adapter whose EventRouter you have modified in the previous steps.
15. Select the Deploy tab in the right pane.

    The right pane displays the following adapter components:

    - BEA_FIX_1_0.rar
    - BEA_FIX_1_0.web.rar

- BEA_FIX_1_0_EventRouter.war.

**Figure 5-5 WebLogic Server Console: Redeploy**



samples> Applications> BEA_FIX_1_0

Connected to pjf-laptop:7001 Active Domain: samples Apr 9, 2003 11:09:28 PM EDT

Edit Application Descriptor...

**Deployment Status by Target:**

| Component | Component Type | Target | Target Type | Deployed | |
|---|---|---|---|---|---|
| BEA_FIX_1_0_EventRouter.war | Web Application | myserver | Server | true | Undeploy Redeploy |
| BEA_FIX_1_0.rar | Connector Component | myserver | Server | true | Undeploy Redeploy |
| BEA_FIX_1_0_Web.war | Web Application | myserver | Server | true | Undeploy Redeploy |

Undeploy Application Undeploy the entire application

Deploy Application Deploy or redeploy the entire application

**Deployment Activity:**

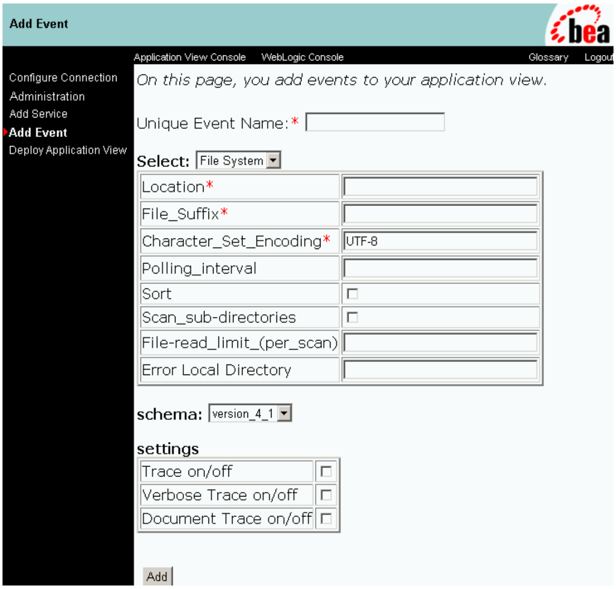
16. Redeploy the EventRouter by clicking the Redeploy button to the right of BEA_FIX_1_0_EventRouter.war.

# Creating Adapter Logs for an Event

To create adapter logs for an event:

1. Create or modify the event.
2. Ensure that all of the adapter parameters are entered correctly.

**Figure 5-6 Add Event window**



*On this page, you add events to your application view.*

Unique Event Name:*

Select: File System

| | |
|---|---|
| Location* | |
| File_Suffix* | |
| Character_Set_Encoding* | UTF-8 |
| Polling_interval | |
| Sort | □ |
| Scan_sub-directories | □ |
| File-read_limit_(per_scan) | |
| Error Local Directory | |

schema: version_4_1

settings

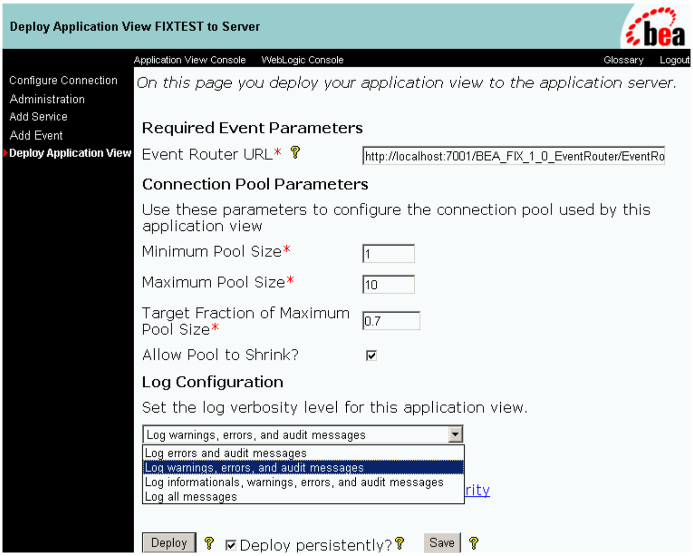
| | |
|---|---|
| Trace on/off | □ |
| Verbose Trace on/off | □ |
| Document Trace on/off | □ |

3. Select the appropriate schema from the drop-down list.
4. Select the appropriate trace levels as described in Table 5-2: Trace, Verbose trace, and Document trace.
5. Click Add to continue to the next configuration pane.
6. Click Continue to move to the next configuration pane.

   The Deploy Application View window opens.
7. Navigate to the Log Configuration area and select the desired trace level.

   This pane enables you to select the trace level for the BEA WebLogic Integration framework.

**Figure 5-7 Deploy Application View window**



Deploy Application View FIXTEST to Server

*On this page you deploy your application view to the application server.*

Required Event Parameters

Event Router URL* http://localhost:7001/BEA_FIX_1_0_EventRouter/EventRo

Connection Pool Parameters

Use these parameters to configure the connection pool used by this application view

Minimum Pool Size* 1

Maximum Pool Size* 10

Target Fraction of Maximum Pool Size* 0.7

Allow Pool to Shrink? ☑

Log Configuration

Set the log verbosity level for this application view.

Log warnings, errors, and audit messages

- Log errors and audit messages
- Log warnings, errors, and audit messages
- Log informationals, warnings, errors, and audit messages
- Log all messages

rity

☑ Deploy persistently?

For maximum tracing, select Log all Messages. This is recommended to obtain optimum debugging information for BEA support personnel.

8. Click Deploy (or Save) to set the trace settings and deploy the application view.

   Traces are created the next time the event occurs.

   Traces are output to a file named `BEA_FIX_1_0.log` in theWebLogic Domain home directory.

# A Supported Messages

This section describes supported messages. It includes the following topic:

- Message Categories, Types, and Descriptions

## Message Categories, Types, and Descriptions

The packaging of FIX application messages in the `manifest.xml` are according the following categories:

**Table A-1 Message Categories, Types, and Descriptions**

| Category | Message Type | Message Descriptions |
|---|---|---|
| Indication | Advertisements | Advertisement messages are used to announce completed transactions. The advertisement message can be transmitted in various transaction types; NEW, CANCEL and REPLACE. |
| | Indication of Interest (IOI) | Indication of interest messages are used to market merchandise, which the broker is buying or selling in either a proprietary or agency capacity. The indications can be time bound with a specific expiration value. Indications are distributed with the understanding that other firms may react to the message first and that the merchandise may no longer be available due to prior trade. Indication messages can be transmitted in various transaction types; NEW, CANCEL, and REPLACE. |

**Table A-1 Message Categories, Types, and Descriptions**

| Category | Message Type | Message Descriptions |
|---|---|---|
| Event Communication | News | The news message is a general free format message between the broker and institution. The message contains flags to identify the news item's urgency and to allow sorting by subject company (symbol). The News message can be originated at either the broker or institution side. |
| | Email | The email message is similar to the format and purpose of the News message, however, it is intended for private use between two parties. |
| Quotation | Quote Request | In some markets it is the practice to request quotes from brokers prior to placement of an order. The quote request message is used for this purpose. This message is commonly referred to as a Request For Quote (RFQ).<br>Quotes can be requested on specific securities or forex rates. The quote request message can be used to request quotes on single products or multiple products.<br>Securities quotes can be requested as either market quotes or for a specific quantity and side. If OrderQty and Side are absent, a market-style quote (bid x offer, size x size) will be returned. |
| | Quote Request Reject | The Quote Request Reject message is used to reject Quote Request messages for all quoting models. |
| | RFQ Request | In tradable and restricted tradable quoting markets, Quote Requests are issued by counterparties interested in ascertaining the market for an instrument. Quote Requests are then distributed by the market to liquidity providers who make markets in the instrument. The RFQ Request is used by liquidity providers to indicate to the market for which instruments they are interested in receiving Quote Requests. It can be used to register interest in receiving quote requests for a single instrument or for multiple instruments. |

**Table A-1 Message Categories, Types, and Descriptions**

| Category | Message Type | Message Descriptions |
|---|---|---|
| | Quote | The quote message is used as the response to a Quote Request message in both indicative, tradable, and restricted tradable quoting markets. In tradable and restricted tradable quoting models, the market maker sends quotes into a market as opposed to sending quotes directly to a counterparty. The quote message can be used to send unsolicited quotes in both indicative, tradable, and restricted tradable quoting markets.<br>The quote message contains a quote for a single product. |
| | Quote Cancel | The Quote Cancel message is used by an originator of quotes to cancel quotes. The Quote Cancel message supports cancellation of:<br>■ All quotes<br>■ Quotes for a specific symbol or security ID<br>■ All quotes for a security type<br>■ All quotes for an underlying<br>Canceling a Quote is accomplished by indicating the type of cancellation in the QuoteCancelType field. |
| | Quote Status Request | The quote status request message is used for the following purposes in markets that employ tradable or restricted tradable quotes:<br>■ For the issuer of a quote in a market to query the status of that quote (using the QuoteID to specify the target quote)<br>■ To subscribe and unsubscribe for Quote Status Report messages for one or more securities |
| | Quote Status Report | The quote status report message is used as:<br>■ the response to a Quote Status Request message<br>■ the response to a Quote Cancel message |

**Table A-1 Message Categories, Types, and Descriptions**

| Category | Message Type | Message Descriptions |
|---|---|---|
| | Mass Quote | The Mass Quote message can contain quotes for multiple securities to support applications that allow for the mass quoting of an option series. Two levels of repeating groups have been provided to minimize the amount of data required to submit a set of quotes for a class of options (for example, all option series for WCOM). |
| | Mass Quote Acknowledgement | Mass Quote Acknowledgement is used as the application level response to a Mass Quote message. |
| Market Data | Market Data Request | Some systems allow the transmission of real-time quote, order, trade and/or other price information on a subscription basis. A Market Data Request is a general request for market data on specific securities or forex quotes. A successful Market Data Request returns one or more Market Data messages containing one or more Market Data Entries. Each Market Data Entry is a Bid, an Offer, a Trade associated with a security, the opening, closing, or settlement price of a security, the buyer or seller imbalance for a security, the value of an index, or the trading session high price, low price, or volume weighted average price (VWAP). |
| | Market Data Snapshot/Full Refresh | Market Data messages can take two forms. The first Market Data message format used for a Snapshot, or a Snapshot + Updates. |
| | Market Data Incremental Refresh | The Market Data message for incremental updates may contain any combination of new, changed, or deleted Market Data Entries, for any combination of instruments, with any combination of trades, imbalances, quotes, index values, open, close, settlement, high, low, and VWAP prices, so long as the maximum FIX message size is not exceeded. All of these types of Market Data Entries can be changed and deleted. |

**Table A-1 Message Categories, Types, and Descriptions**

| Category | Message Type | Message Descriptions |
|---|---|---|
| | Market Data Request Reject | The Market Data Request Reject is used when the broker cannot honor the Market Data Request, due to business or technical reasons. Brokers may choose to limit various parameters, such as the size of requests, whether just the top of book or the entire book may be displayed, and whether Full or Incremental updates must be used. |
| Security and Trading Definition/Status | Security Definition Request | The Security Definition Request message is used to request a specific Security to be traded with the second party. The request security can be defined as a multileg security made up of one or more instrument legs. |
| | Security Definition | The Security Definition message is used for the following:<br>■ Accept the security defined in a *Security Definition* message<br>■ Accept the security defined in a *Security Definition* message with changes to the definition and/or identity of the security<br>■ Reject the security requested in a *Security Definition* message |
| | Security Type Request | The Security Type Request message is used to return a list of security types available from a counterparty or market. |
| | Security Types | The Security Type message is used to return a list of security types available from a counterparty or market. |
| | Security List Request | The Security List Request message is used to return a list of securities from the counterparty that match criteria provided on the request. |
| | Security List | The Security List message is used to return a list of securities that matches the criteria specified in a Security List Request. |
| | Derivative Security List Request | The Derivative Security List Request message is used to return a list of securities from the counterparty that match criteria provided on the request. |

Table A-1 Message Categories, Types, and Descriptions

| Category | Message Type | Message Descriptions |
|---|---|---|
| | Derivative Security List | The Derivative Security List message is used to return a list of securities that matches the criteria specified in a Derivative Security List Request. |
| | Status Security Request | The Security Status Request message provides for the ability to request the status of a security. One or more Security Status messages are returned as a result of a Security Status Request message. |
| | Security Status | The Security Status message provides for the ability to report changes in status to a security. The Security Status message contains fields to indicate trading status, corporate actions, financial status of the company. The Security Status message is used by one trading entity (for instance an exchange) to report changes in the state of a security. |
| | Trading Session Status Request | The Trading Session Status Request is used to request information on the status of a market. With the move to multiple sessions occurring for a given trading party (morning and evening sessions for instance) there is a need to be able to provide information on what product is trading on what market. |
| | Trading Session Status | The Trading Session Status provides information on the status of a market. With the move to multiple sessions occurring for a given trading party (morning and evening sessions for instance) there is a need to be able to provide information on what product is trading on what market. |
| Single/General Order Handling | New Order | The new order message type is used by institutions wishing to electronically submit securities and forex orders to a broker for execution. The New Order message type may also be used by institutions or retail intermediaries wishing to electronically submit Collective Investment Vehicle (CIV) orders to a broker or fund manager for execution. |

**Table A-1 Message Categories, Types, and Descriptions**

| Category | Message Type | Message Descriptions |
|---|---|---|
| | Execution Report | The execution report message is used to:<br>■ Confirm the receipt of an order<br>■ Confirm changes to an existing order (that is, accept cancel and replace requests)<br>■ Relay order status information<br>■ Relay fill information on working orders<br>■ Relay fill information on tradable or restricted tradable quotes<br>■ Reject orders<br>■ Report post-trade fees calculations associated with a trade |
| | Don’t Know Trade (DKT) | The Don’t Know Trade (DK) message notifies a trading partner that an electronically received execution has been rejected. This message can be thought of as an execution reject message. |
| | Order/Cancel/Replace Request | The order cancel/replace request is used to change the parameters of an existing order. |
| | Order Cancel Request | The order cancel request message requests the cancellation of all of the remaining quantity of an existing order. |
| | Order Cancel Reject | The order cancel reject message is issued by the broker upon receipt of a cancel request or cancel/replace request message, which cannot be honored. Requests to change price or decrease quantity are executed only when an outstanding quantity exists. Filled orders cannot be changed (for example, quantity reduced or price change. However, the broker/sellside may support increasing the order quantity on a currently filled order). |
| | Order Status Request | The order status request message is used by the institution to generate an order status message back from the broker. |

**Table A-1 Message Categories, Types, and Descriptions**

| Category | Message Type | Message Descriptions |
|---|---|---|
| | Order Mass Cancel Request | The order mass cancel request message requests the cancellation of all of the remaining quantity of a group of orders matching criteria specified within the request. NOTE: This message can only be used to cancel messages (reduce the full quantity). |
| | Order Mass Cancel Report | The Order Mass Cancel Report is used to acknowledge an Order Mass Cancel Request. Note that each affected order that is canceled is acknowledged with a separate Execution Report or Order Cancel Reject message. |
| | Order Mass Status Request | The order mass status request message requests the status for orders matching criteria specified within the request. |
| Cross Orders | New Order - Cross | Used to submit a cross order into a market. The cross order contains two order sides (a buy and a sell). |
| | Cross Order Cancel/Replace Request | Used to modify a cross order previously submitted using the New Order - Cross message. See Order Cancel Replace Request for details concerning message usage. Refer to the Order Cancel Replace Request (a.k.a. Order Modification Request) message for restrictions on what fields can be changed during a cancel replace. |
| | Cross Order Cancel Request | Used to fully cancel the remaining open quantity of a cross order. |

**Table A-1 Message Categories, Types, and Descriptions**

| Category | Message Type | Message Descriptions |
|---|---|---|
| Multi Leg Orders | New Order - Multileg | The New Order - Multileg is provided to submit orders for securities that are made up of multiple securities, known as legs. Swaps, option strategies, futures spreads, are a few examples of multileg securities. A multileg security is made up of multiple securities that are traded atomically. This requirement that all legs be traded in the quantities that they make up the multlileg security is the important distinction between a multileg order and a list order.<br>Two generalized approaches to trading multileg securities are supported by FIX. The first approach involves a market maintaining multileg securities as separate products for which markets can be created. This “product approach” is often used in electronic trading systems. The second approach is to trade the multileg security as a group of separate securities – as is commonly done today in open outcry markets. |
|  | Multi-leg Order Cancel/Replace Request | Used to modify a multileg order previously submitted using the New Order - Multileg message. See Order Cancel Replace Request for details concerning message usage. |
| List Program Basket Trading | Bid Request | The BidRequest Message can be used in one of two ways depending on which market conventions are being followed. In the “Non disclosed” convention (for example, US/European model) the BidRequest message can be used to request a bid based on the sector, country, index and liquidity information contained within the message itself. In the “Non disclosed” convention the entry repeating group is used to define liquidity of the program. In the “Disclosed” convention (for example, Japanese model) the BidRequest message can be used to request bids based on the ListOrderDetail messages sent in advance of BidRequest message. In the “Disclosed” convention the list repeating group is used to define which ListOrderDetail messages a bid is being sort for and the directions of the required bids. |

**Table A-1 Message Categories, Types, and Descriptions**

| Category | Message Type | Message Descriptions |
|---|---|---|
| | Bid Response | The Bid Response message can be used in one of two ways depending on which market conventions are being followed:<br>■ In the "Non disclosed" convention the Bid Response message can be used to supply a bid based on the sector, country, index and liquidity information contained within the corresponding bid request message<br>■ In the "Disclosed" convention the Bid Response message can be used to supply bids based on the List<br>Order Detail messages sent in advance of the corresponding Bid Request message. |
| | New Order - List | The NewOrderList Message can be used in one of two ways depending on which market conventions are being followed:<br>■ In the "Non disclosed" convention the New Order - List message is sent after the bidding process has been completed, by telephone or electronically. The New Order - List message enumerates the stocks, quantities, direction for the trade and may contain pre-allocation information. This message may also be used as the first message for the transmission of a program trade where the bidding process has been done by means other than FIX. In this scenario the messages may either be used as a staging process, in which case the broker will start execution once either a ListExecute is received or for immediate execution, in which case the orders will be executed on receipt.<br>■ In the "Disclosed" convention the New Order - List message is sent before the bidding process is started, by telephone or electronically. The New Order - List message enumerates the stocks and quantities from the bidding process, and may contain pre-allocation information. The direction of the trade is disclosed after the bidding process is completed. |

**Table A-1 Message Categories, Types, and Descriptions**

| Category | Message Type | Message Descriptions |
|---|---|---|
| | List Strike Price | The strike price message is used to exchange strike price information for principal trades. It can also be used to exchange reference prices for agency trades. |
| | List Status | The list status message is issued as the response to a List Status Request message sent in an unsolicited fashion by the sell-side. It indicates the current state of the orders within the list as they exist at the broker's site.<br>Orders within the list are statused at the summary level. Individual executions are not reported, rather, the current state of the order is reported. |
| | List Execute | The list execute message type is used by institutions to instruct the broker to begin execution of a previously submitted list. This message may or may not be used, as it may be mirroring a phone conversation. |
| | List Cancel Request | The list cancel request message type is used by institutions wishing to cancel previously submitted lists either before or during execution. |
| | List Status Request | The list status request message type is used by institutions to instruct the broker to generate status messages for a list. |

Table A-1 Message Categories, Types, and Descriptions

| Category | Message Type | Message Descriptions |
|---|---|---|
| Allocation and Ready To Book | Allocation | The Allocation message provides the ability to specify how an order or set of orders should be subdivided amongst one or more accounts. It can also be used as a confirmation message through which third parties can communicate execution and settlement details between trading partners. In addition, the allocation message can be sent by the broker to communicate fees and other details that can only be computed once the sub-account breakdowns are known. Note the response to the Allocation message is the AllocationACK message.<br>The Allocation message can also be sent by the buyside firm after execution to indicate to the sellside firm that one or a combined (aggregated) set of orders are "Ready-To-Book" without specifying individual account breakdowns. This can be used to trigger post-trade allocation, matching, and settlement processing via other channels (for example, post-trade industry utilities). |
| | Allocation ACK | The Allocation ACK message is used to acknowledge the receipt and status of an Allocation message.<br>It is possible that multiple Allocation ACK messages can be generated for a single allocation to detail the receipt and then the acceptance or rejection of the Allocation message. |
| Settlement Instructions | Settlement Instructions | The Settlement Instructions message provides the broker’s, the institution’s, or the intermediary’s instructions for trade settlement. The SettlInstSource field indicates if the settlement instructions are the broker’s, the institution’s, or the intermediary’s. This message has been designed so that it can be sent from the broker to the institution, from the institution to the broker, or from either to an independent “standing instructions” database or matching system or, for CIV, from an intermediary to a fund manager. |

**Table A-1 Message Categories, Types, and Descriptions**

| Category | Message Type | Message Descriptions |
|---|---|---|
| Trade Capture Reporting | Trade Capture Report Request | The Trade Capture Report can be used to:<br>■ Request one or more trade capture reports based upon selection criteria provided on the trade capture report request<br>■ Subscribe for trade capture reports based upon selection criteria provided on the trade capture report request |
| | Trade Capture Report | The Trade Capture Report message can be:<br>■ Used to report trades between counterparties<br>■ Can be sent unsolicited between counterparties<br>■ Sent as a reply to a Trade Capture Report Request<br>■ Can be used to report unmatched and matched trades |
| Registration Instructions | Registration Instructions | The Registration Instructions message type may be used by institutions or retail intermediaries wishing to electronically submit registration information to a broker or fund manager (for CIV) for an order or for an allocation.<br>A Registration Instructions message can be submitted as new, cancel or replace. The RegistTransType field indicates the purpose of the message. |
| | Registration Instructions Response | The Registration Instructions Response message type may be used by broker or fund manager (for CIV) in response to a Registration Instructions message submitted by an institution or retail intermediary for an order or for an allocation. |